TUGAS AKHIR – SM141501

APLIKASI METODE PARTICLE FILTER PADA PELACAKAN SERTA PENGHITUNGAN ORANG

MUHAMMAD SIFA'UL RIZKY
NRP 1213 100 067

Dosen Pembimbing
Dr. Dwi Ratna Sulistyaningrum, S.Si, MT

DEPARTEMEN MATEMATIKA
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – SM141501

APPLICATION OF PARTICLE FILTER METHOD IN PEOPLE TRACKING AND COUNTING

MUHAMMAD SIFA'UL RIZKY
NRP 1213 100 067

Supervisor
Dr. Dwi Ratna Sulistyaningrum, S.Si, MT

DEPARTMENT OF MATHEMATICS
Faculty of Mathematics and Natural Science
Institut Teknologi Sepuluh Nopember
Surabaya 2017

LEMBAR PENGESAHAN

APLIKASI METODE PARTICLE FILTER PADA PELACAKAN SERTA PENGHITUNGAN ORANG

APPLICATION OF PARTICLE FILTER METHOD IN PEOPLE TRACKING AND COUNTING

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
Untuk memperoleh gelar Sarjana Sains
Pada bidang minat Ilmu Komputer
Program Studi S-1 Departemen Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember Surabaya

Oleh :
MUHAMMAD SIFA'UL RIZKY
NRP. 1213 100 067

Menyetujui,
Dosen Pembimbing

Dr. Dwi Ratna Sulistyaningrum, S.Si,MT
NIP. 19690405 199403 2 003

Mengetahui,
Kepala Departemen Matematika
FMIPA ITS

Dr. Imam Mukhlash, S.Si, MT
NIP. 19700831 199403 1 003
Surabaya, 2 Agustus 2017

APLIKASI METODE *PARTICLE FILTER* PADA PELACAKAN SERTA PENGHITUNGAN ORANG

Nama Mahasiswa	: Muhammad Sifa'ul Rizky
NRP	: 1213 100 067
Departemen	: Matematika
Dosen Pembimbing	: Dr. Dwi Ratna S, S.Si,MT

ABSTRAK

Teknologi berkembang cukup pesat selama beberapa tahun terakhir, terutama dalam bidang pelacakan objek. Terlebih lagi terhadap objek yang diteliti yaitu orang, apalagi dalam jumlah yang banyak. Penelitian ini bertujuan untuk menerapkan metode *Particle Filter* untuk melakukan pelacakan serta penghitungan orang yang berada dalam *area* tertentu. Pelacakan orang menjadi sedikit sulit dengan adanya beberapa halangan, salah satunya adalah oklusi. Oklusi adalah gangguan dimana membuat keakuratan suatu sistem berkurang. Metode *Particle Filter* menggunakan estimasi gerak yang ditambahkan pada model yang digunakan. Estimasi gerakan objek dilakukan dengan metode *Hierarchical Block Matching*. Selanjutnya dilakukan pelacakan objek yang diawali dengan membangkitkan sejumlah partikel dengan bobot awal, *update* bobot, *resample* dan *update* posisi objek sehingga diperoleh posisi objek di setiap *frame*. Kemudian setelah didapatkan posisi objek maka dapat dilakukan penghitungan objek yang berada dalam *area* yang telah ditentukan. Pada tugas akhir ini telah dikerjakan pelacakan dan penghitungan objek agar dapat menjadi satu kesatuan dan dapat mengatasi oklusi. Setelah pengujian program didapatkan hasil bahwa penghitungan orang dapat dilakukan dengan baik dengan akurasi rata-rata sebesar 86,7 % , sedangkan untuk pelacakan orang belum dapat melacak ketika terjadi oklusi keseluruhan, hanya bisa melacak ketika oklusi sebagian saja.

Kata Kunci : particle filter, pelacakan orang, penghitungan orang.

APPLICATION OF PARTICLE FILTER METHOD IN PEOPLE TRACKING AND COUNTING

Name of Student	: Muhammad Sifa'ul Rizky
NRP	: 1213 100 067
Department	: Mathematics
Supervisor	: Dr. Dwi Ratna S, S.Si,MT

ABSTRACT

Technology has grown quite rapidly over the past few years, especially in the field of object tracking. Moreover, the object tracking is people, especially in large quantities. This study aims to apply Particle Filter method to track and calculate people who are in certain areas. Tracking people becomes a little difficult with some obstacles, for example is occlusion. Occlusion is a disorder which makes the accuracy of a system diminish. The Particle Filter method uses the estimated motion added to the model used. Estimation of object movement is done by Hierarchical Block Matching method. Furthermore, object tracking begins with generating a number of particles with initial weights. Update the weight, resample and update the position of the object so that the position of objects in each frame. Then after obtaining the position of the object then can be calculated objects that are within the area that has been determined. In this final project has been done tracking and counting object in order to become one unity and can overcome occlusion. After the test the results obtained that the calculation of people can be done well with an average accuracy of 86.7%, while for tracking people have not been able to track when the occurrence of the whole occlusion, can only track when only partial occlusion.

Keywords : particle filter, object tracking, people counting.

KATA PENGANTAR

Segala Puji bagi Allah SWT Tuhan semesta alam yang telah telah memberikan karunia, rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul: **"Aplikasi Metode Particle Filter Pada Pelacakan Serta Penghitungan Orang"** yang merupakan salah satu persyaratan akademis dalam menyelesaikan Program Studi S-1 pada Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dapat diselesaikan dengan berkat kerjasama, bantuan, dan dukungan dari banyak pihak. Sehubungan dengan hal itu, penulis mengucapkan terima kasih kepada:

1. Dr. Dwi Ratna Sulistyaningrum, S.Si, MT selaku dosen pembimbing yang senantiasa membimbing dengan sabar dan memberikan kritik dan saran dalam penyusunan Tugas Akhir ini.
2. Dr. Imam Mukhlash, S.Si, MT selaku Kepala Departemen Matematika.
3. Drs. Iis Herisman, M.Sc.. selaku Dosen Wali
4. Prof. DR. Mohammad Isa Irawan, MT, Drs. Soetrisno, MI.Komp., Sunarsini, S.Si, M.Si, dan Dra. Sri Suprapti Hartatiati, M.Si selaku dosen penguji Tugas Akhir ini.
5. Dr. Didik Khusnul Arif, S.Si., M.Si.. selaku Koordinator Tugas Akhir.
6. Seluruh jajaran dosen dan staf Departemen Matematika ITS.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan saran dan kritik dari pembaca. Akhir kata, semoga Tugas Akhir ini bermanfaat bagi semua pihak yang berkepentingan.

Surabaya, Agustus 2017

Penulis

special thanks to:

Selama proses pembuatan Tugas Akhir ini, banyak pihak yang telah memberikan bantuan dan dukungan untuk penulis. Penulis mengucapkan terima kasih dan apresisasi secara khusus kepada:

1. Ayah Djumadi Susilo dan Ibu Hartatik tersayang yang senantiasa dengan ikhlas memberikan semangat, perhatian, kasih sayang, doa, motivasi dan nasihat yang sangat berarti bagi penulis.
2. Adik Sovina Dhiya' Ulhaq yang ikut memotivasi bagi penulis untuk segera menyelesaikan Tugas Akhir ini.
3. Teman-teman pejuang 116 yang tidak bisa disebutkan satu per satu yang telah bersama-sama penulis berjuang dalam suka dan duka.
4. Sahabat Matematika 2013 yang tidak bisa disebutkan satu persatu yang tidak bosan dalam memberi semangat dan saran untuk menyelesaikan Tugas Akhir ini.
5. Teman-teman seperti Ivan, Agus, Romli, Bayu, Fadhlan, Firdo, Amel, Brigita, Wawan, Gery, Dita, Didit, Syahira, dan yang lainnya yang telah ikut membantu penulis dalam menyelesaikan Tugas Akhir ini.
6. Saudari Fina Nurul Atikasari yang selalu menyemangati dan memberi do'a dan dukungannya kepada penulis.

Tentu saja masih banyak pihak lain yang turut andil dalam penyelesaian tugas akhir ini yang tidak bisa penulis sebutkan satu persatu. Semoga Allah membalas dengan balasan yang lebih baik bagi semua pihak yang telah membantu penulis. *Aamiin yaa rabbal 'aalamiin*.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

Pada bab ini dibahas mengenai latar belakang yang mendasari penulisan Tugas Akhir ini. Di dalamnya mencakup identifikasi permasalahan pada topik Tugas Akhir kemudian dirumuskan menjadi permasalahan yang diberikan batasan-batasan dalam pembahasan pada Tugas Akhir ini.

1.1 Latar Belakang Masalah

Perkembangan teknologi telah meningkat cukup pesat dalam beberapa tahun terakhir. Banyak sekali aplikasi yang telah dikembangkan atau telah digunakan dalam kehidupan, baik itu untuk kepentingan pribadi ataupun umum. Salah satunya adalah kamera pengawas yang sering digunakan untuk mengawasi kegiatan orang, entah itu ditempatkan di jalan raya, di tempat parkir, di kantor. Kamera pengawas biasanya selalu dilengkapi dengan pelacakan objek untuk mengetahui pergerakan objek. Pelacakan objek visual telah menarik perhatian yang cukup besar karena banyak sekali kejadian yang memerlukan pelacakan objek, baik itu pendeteksian objek, menganalisa aktivitas orang, interaksi antar orang dengan komputer, dan sebagainya. Sebagai contoh pengawasan di dalam lingkungan sekolah , lalu lintas, kantor dan lain-lain [1].

Pelacakan objek sendiri merupakan salah satu cara untuk memudahkan pekerjaan orang. Biasanya pelacakan serta penghitungan objek, terutama orang digunakan untuk memonitor aktivitas yang ada di lingkungan tersebut [2]. Pelacakan objek sendiri dapat berupa pelacakan satu objek atau pelacakan banyak objek, tergantung objek yang akan diteliti [3]. Pelacakan satu objek atau beberapa objek relatif sederhana dari pelacakan beberapa objek dengan kondisi latar belakang yang buruk [4]. Namun dalam

prakteknya, pengaturan untuk mengambil video dari lokasi yang sama untuk kamera yang berbeda mungkin tidak selalu dapat digunakan. Yang tersedia saat ini adalah pendekatan secara tunggal yang dapat menangani objek yang terisolasi secara efektif, sementara pelacakan beberapa objek yang sangat terhambat oleh gangguan dan mungkin sering gagal, dengan adanya oklusi.

Oklusi telah dianggap sebagai salah satu rintangan yang menantang dalam pelacakan objek visual karena mengurangi akurasi pelacakan. Dalam citra video, beberapa bagian dari objek mungkin tidak terlihat karena oklusi. Seorang orang dapat mengenali suatu objek meskipun sebagian terkena oklusi. Jika objek adalah sebagian terlihat, otak orang dapat merekonstruksi seluruh objek menggunakan inferensi berdasarkan bagian terlihat dari objek dan pengetahuan struktur umum objek.

Ketika terjadi oklusi, teknik yang canggih mungkin diperlukan untuk melaksanakan pelacakan sistem yang mendekati mekanisme pengenalan objek dari orang. Banyak metode pelacakan yang ada menunjukkan pelacakan yang baik untuk beberapa objek bergerak ketika objek yang jelas terpisah satu sama lain dan warna mereka berbeda dari orang-orang dari latar belakang. Namun, metode pelacakan ini mungkin dinyatakan gagal dan pelacakan objek dapat beralih ke yang lain objek bergerak atau ke lokasi di suatu tempat di latar belakang. Tujuan dari penelitian ini adalah menemukan cara untuk menangani masalah tersebut.

Selain itu, penghitungan objek juga berperan dalam sistem keamanan untuk memudahkan pengawasan terutama apabila ada hal-hal yang tidak diinginkan ataupun untuk keperluan yang lain. Seperti halnya penghitungan jumlah orang yang memasuki sebuah gedung, dengan sensor infra merah. Walaupun terlihat sederhana, namun sebenarnya lebih efisien apabila tidak ada dua orang yang

bergerak mengenai sensor tersebut. Oleh karena itu dibutuhkan suatu sistem yang dapat menghitung jumlah orang yang berada di *area* yang telah ditentukan serta melacaknya.

Dalam Tugas Akhir ini akan diterapkan sebuah metode yang dapat digunakan untuk melacak orang dalam hal ini adalah orang serta menghitungnya. Akan digunakan metode *particle filter* yang berfungsi untuk melacak orang yang berada dalam *area* serta menghitung jumlahnya.

1.2 Rumusan Masalah

Rumusan masalah untuk tugas akhir ini adalah sebagai berikut:

1. Bagaimana melacak orang menggunakan metode *particle filter*?
2. Bagaimana kinerja metode *particle filter* untuk pelacakan dengan adanya oklusi serta penghitungan orang?

1.3 Batasan Masalah

Batasan-batasan yang ada dalam pembuatan Tugas Akhir ini adalah :

1. Video yang digunakan berasal dari kamera yang statis.
2. Video diambil dari rekaman luar ruangan.
3. Jumlah orang yang dihitung adalah yang berada dalam area ROI

1.4 Tujuan Penelitian

Tujuan dari penulisan ini adalah:

1. Menerapkan pelacakan orang dengan menggunakan metode *particle filter*.
2. Mengetahui kinerja dari metode *particle filter* untuk pelacakan dengan adanya oklusi serta penghitungan orang yang berada dalam ROI.

1.5 Manfaat Penelitian

Manfaat dari penulisan ini adalah:

1. Untuk keperluan keamanan, cukup membantu melacak serta menghitung orang-orang yang berada di *area* tersebut.
2. Untuk mengamati tingkah laku orang, entah itu berlari, berjalan dan lain-lain yang berada dalam suatu *area* tertentu.
3. Untuk menganalisa *area* mana saja yang sering dikunjungi agar dapat dilakukan optimasi terhadap perilaku orang-orang.

1.6 Sistematika Penulisan

Sistematika Penulisan Tugas Akhir ini dibagi menjadi enam bab. Isi dari masing- masing bab tersebut adalah sebagai berikut :

1. Bab I Pendahuluan
   Pada bab ini dibahas tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat dari penelitan yang dilakukan, serta sistematika penulisan Tugas Akhir.
2. Bab II Dasar Teori
   Pada bab ini diuraikan beberapa teori pendukung yang berasal dari jurnal yang berkaitan atau yang melandasi pembahasan pada penelitian ini untuk membantu menyelesaikan permasalahan Tugas Akhir.
3. Bab III Metodologi
   Pada bab ini berisi tentang langkah-langkah yang digunakan untuk menyelesaikan Tugas Akhir.
4. Bab IV Perancangan dan Implementasi Sistem
   Bab ini menjelaskan tentang pengambilan data rekaman diluar ruangan. Kemudian juga dijelaskan mengenai perancangan sistem yang berisi aturan main berjalannya program dari awal hingga akhir. Bab ini juga menjelaskan tentang proses pengolahan informasi video sehingga dapat digunakan untuk melacak dan menghitung jumlah orang.

5. Bab V Uji Coba dan Pembahasan
   Bab ini menjelaskan tentang uji coba program yang telah dibuat, kemudian dilihat akurasinya. Setelah itu akan dilakukan pembahasan mengenai hasil uji coba yang telah dilakukan untuk bahan pertimbangan kedepannya.
6. Bab VI Penutup
   Bab ini merupakan penutup, berisi tentang kesimpulan yang dapat diambil berdasarkan hasil uji coba dan saran yang selayaknya dilakukan bila Tugas Akhir ini dilanjutkan.

BAB II
DASAR TEORI

Bab ini menjelaskan tentang kajian teori dari referensi penunjang serta penjelasan permasalahan yang dibahas dalam tugas akhir ini, meliputi Penelitian Sebelumnya dan beberapa dasar-dasar teori yang akan dijelaskan pada sub bab selanjutnya.

2.1 Penelitian Sebelumnya

Sudah banyak penelitian yang berkaitan dengan pelacakan, entah itu satu objek atau banyak objek. Penelitian tersebut bertujuan untuk mendapatkan pelacakan yang efektif. Entah itu dengan menggunakan deteksi objek, atau algoritma yang berbeda seperti *Kalman filter, mean shift* dan *particle filter* [5]. Dalam pelacakan orang, Jin et al. [6] membagi bagian tubuh menjadi 3, representasi dari kombinasi antara histogram warna dan *HOG*, serta dilacak secara terpisah. Namun metode tersebut hanya melacak objek secara tunggal dan tidak bisa mengatasi oklusi. Chang dan Ansari [7] menggunakan model elliptik dan estimasi gradien. Walaupun *kernel particle filter* yang dirumuskan menunjukkan hasil yang lebih baik dari *particle filter* yang biasa, namun tidak dapat melacak banyak objek dan tidak dapat mengatasi oklusi. Cabido et al. [8] merumuskan algoritma pelacakan yang mengkombinasikan *particle filter* dengan algoritma *memetic*. Walaupun dapat melacak banyak objek, tetapi belum dapat menangani oklusi. Liu dan Sun [9] menggunakan *particle filter* untuk melacak objek yang direpresentasikan oleh kotak. Performa pelacakan meningkat karena penghitungan dilakukan antara kombinasi dari histogram dan kesamaan *Bhattacharyya*. Metode ini menghasilkan proses yang cepat namun kurang baik dalam akurasi. Yang et al. [10] menggunakan metode *particle filter* untuk pelacakan banyak objek dengan quasi sample acak. Kemudian Cai et al. [11] menggunakan *particle filter* yang

dikombinasikan dengan algoritma *mean shift* namun kelemahannya adalah banyak partikel yang dibutuhkan untuk training.

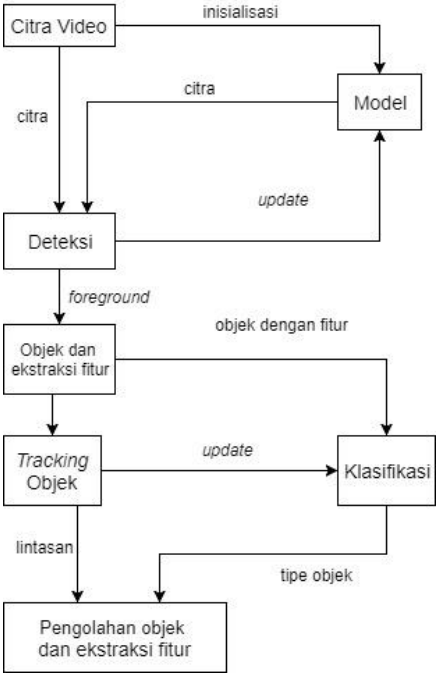
2.2 Tracking

Tracking atau pelacakan secara harfiah memiliki arti mengikuti jalan, atau dalam arti bebasnya adalah proses mencari objek bergerak dalam urutan *frame* yang dikenal sebagai pelacakan. Pelacakan ini dapat dilakukan dengan menggunakan ekstraksi ciri benda dan mendeteksi objek/benda bergerak diurutan *frame*. Dengan menggunakan nilai posisi objek di setiap *frame*, bisa digunakan untuk menghitung posisi dan kecepatan objek bergerak [8].

Pelacakan objek bergerak atau *object tracking* adalah proses mencari lokasi dari objek yang akan diamati dari suatu data video untuk setiap satu satuan *frame* dalam data video tersebut. Pelacakan objek bergerak dibutuhkan dalam banyak aplikasi visual. Beberapa diantaranya adalah pengaturan lalu lintas, robot vision, pengembangan militer, sports analysis dan sistem pengawasan/pengamanan pada wilayah tertentu, misalnya tempat parkir, mall dll. Gambar 2.1 adalah sebuah ilustrasi yang menggambarkan adanya suatu pelacakan objek bergerak.

Berbagai metode dikembangkan untuk menyelesaikan permasalahan pelacakan objek bergerak antara lain *SIFT, Mean Shift, CAMSHIFT, Kalman Filter, Extended Kalman Filter* dan *Particle Filter*. *SIFT, Mean Shift* dan *CAMSHIFT* melakukan pelacakan objek berdasarkan pengenalan objek. Sedangkan *Kalman Filter, Extended Kalman Filter* dan *Particle Filter* melakukan pelacakan objek berdasarkan gerakan objek.

Pada pelacakan objek bergerak terdapat dua pendekatan yaitu *tracking-by-detection* dan *detection-by-tracking*. Pendekatan pertama (*tracking-by-detection*), melakukan pelacakan objek dengan cara mendeteksi objek pada setiap *frame* video yang diamati, guna menentukan posisi objek itu di setiap *frame*nya. Pendekatan kedua (*detection-by-tracking*), melakukan pelacakan objek dengan cara mendeteksi objek hanya pada satu atau beberapa *frame* pertama saja, lalu untuk mengetahui gerak atau posisi objek pada *frame-frame* selanjutnya cukup melakukan pelacakan perubahan gerak antara *frame* saat ini terhadap *frame-frame* sebelumnya. Pendekatan yang pertama tidak dianjurkan selain karena mahalnya biaya komputasi untuk mendeteksi keberadaan suatu objek pada setiap *frame*nya, pendekatan ini juga tidak memanfaatkan informasi yang sudah didapat pada *frame-frame* sebelumnya yang dapat digunakan untuk menghitung probabilitas kemunculan atau posisi suatu objek saat ini.



Gambar 2.1. Blok Diagram Pelacakan Objek Bergerak

Gambar 2.1 menunjukkan diagram pelacakan objek bergerak yang umum digunakan dalam pendeteksian dan pelacakan objek. Pada bagian awal dilakukan segmentasi untuk memisahkan *background* dan *foreground*. *Background subtraction* dan *foreground detection* dapat digunakan untuk mendapatkan objek dan dapat mengekstraksi atribut objek untuk selanjutnya dilakukan deteksi objek dan ekstraksi fitur. Deteksi objek digunakan dengan mengetahui perbedaan *background* dengan *foreground*. Ekstraksi fitur yang tersedia biasanya berupa histogram, *wavelet*, fitur berdasarkan warna, dan lain-lain. Ekstraksi fitur yang digunakan dalam Tugas Akhir ini adalah menggunakan ekualisasi histogram, untuk meratakan derajat

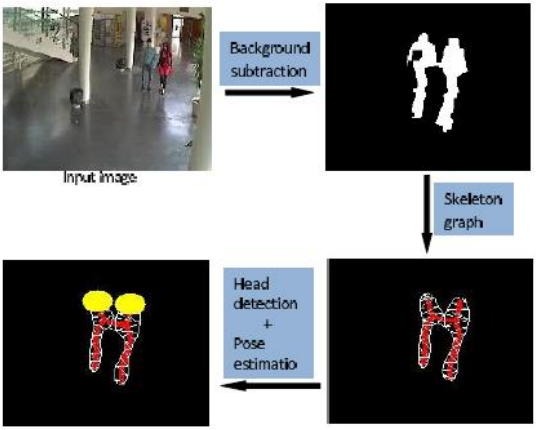
keabuan. Setelah itu pelacakan objek dapat dilakukan dan untuk objek dengan fitur yang telah diekstrak dapat dilakukan klasifikasi objek. Namun pada Tugas Akhir ini hanya melakukan pelacakan objek bergerak dan tidak melakukan klasifikasi terhadap objek tersebut.

2.3 Penghitungan Orang (*People Counting*)

People counting adalah perhitungan objek yang bergerak berupa orang yang melintasi bagian tertentu pada waktu tertentu. *People counting* sendiri telah banyak digunakan untuk menganalisa orang-orang yang berada dalam sebuah ruangan, misalnya saja di dalam mall, gedung bertingkat, dan lain-lain. *People counting* biasanya digunakan dalam area tertentu saja.[2] Banyak cara digunakan untuk melakukan people counting. Salah satu caranya adalah dengan mendeteksi orang yang berlalu-lalang dengan garis virtual sebagai penandanya. Apabila ada orang yang masuk dari sebelah kiri kemudian bergerak melewati garis tersebit, maka akan dihitung, hal yang sama juga berlaku untuk sebaliknya. Ketika ada orang yang melewati garis tersebut,maka akan dihitung. Namun akan ada masalah bila sistem dilakukan seperti itu yaitu bila objek yang bergerak bukan hanya orang, dengan kondisi setiap objek yang melalui garis virtual akan dihitung maka akan menimbulkan *error*. Oleh karena itu dibutuhkan pengenalan bahwa orang yang bergerak berbeda dengan objek lain yang bergerak. Metode yang digunakan untuk proses *counting* juga bermacam-macam, salah satu yang sering digunakan adalah HOG (*Histogram of Oriented Gradients*). Sistem seperti ini sebenarnya sederhana karena hanya mengandalkan orang yang lewat, namun terkadang ada kondisi dimana ada dua orang yang melewati garis tersebut, namun hanya dianggap satu orang saja. Apalagi bila sistem tersebut ditempatkan pada *area* pusat perbelanjaan yang biasanya cukup padat akan pengunjung. Oleh karena itu *people counting* ini mulai dikembangkan lebih luas lagi.

Biasanya *people counting* juga digunakan untuk memprediksi jumlah pejalan kaki yang lewat, disertai dengan arah

kemana mereka lewat, dari satu titik ke titik tertentu. Dari sistem *people counting* sendiri digunakan untuk mengurangi biaya pengawasan dimana biasanya pengawasan melibatkan tenaga manusia.[18] Informasi berupa pergerakan orang-orang dari satu titik ke titik lain dapat digunakan untuk berbagai macam aplikasi seperti dalam pameran dan sebagainya. Dari situ didapatkan data bahwa dalam hal ini pengunjung lebih senang berada dalam area mana, dan bisa sebagai acuan agar penempatannya lebih mudah dijangkau oleh pengunjung.



Gambar 2.2. Sistem *people counting* [18]

Langkah-langkah *people counting* adalah sebagai berikut:

1. Pilih *area* ROI dari *frame*
2. Lakukan proses *tracking* dengan metode sebelumnya untuk mendeteksi orang yang sedang bergerak.
3. Dari proses tracking didapatkan *centroid* dari masing-masing objek yang akan dignakan untuk menghitung orang yang berada dalam *area* ROI.

4. Ketika *centroid* memasuki *area* yang telah ditetapkan, maka orang tersebut akan dihitung, apabila *centroid* berada diluar *area* maka tidak akan dihitung.

Dalam prakteknya, *people counting* digunakan untuk mengestimasi jumlah orang yang berada dalam suatu *area*, entah dengan garis virtual ataupun dengan sebuah *area* sebagai pendeteksian. Ada banyak metode untuk mendeteksi perpindahan orang yang bergerak yaitu *Gaussian Mixture Model, Background Substraction, Neural Networks*, dan lain-lain.[18]

2.4 Metode *Gaussian Mixture Model*

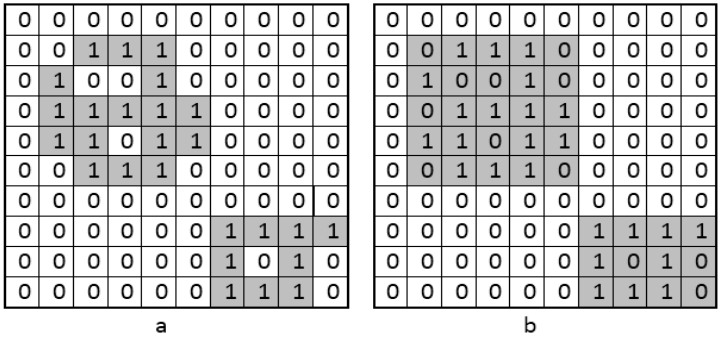
Gaussian Mixture Model (GMM) merupakan metode yang tepat untuk berbagai kondisi yang terdapat pada citra, seperti background citra yang selalu statis, multimodal, maupun yang mengandung noise (gangguan atau objek yang tidak diinginkan terdapat pada citra). Pada proses GMM dibutuhkan algoritma *clustering* untuk mengelompokkan pixel-pixel mana saja yang termasuk *foreground* atau *background*. Model-model GMM terbentuk dari data warna pixel berdasarkan waktu. Hasil model tersebut akan menjadi 2 bagian, model yang mencerminkan *background* dan model non-*background*. Jumlah model GMM yang digunakan mempengaruhi jumlah model *background*. Semakin besar jumlah model GMM yang dipakai semakin banyak model background yang dimiliki suatu piksel. R. Arif telah melakukan Uji coba terhadap parameter-parameter Gaussian menggunakan video rekaman yang berdurasi 30 detik. Sehingga diperoleh kesimpulan pada penelitiannya yaitu dengan menggunakan jumlah model antara 3 s.d 7 dan *learning rate* 0.001, 0.005, atau 0.01 dapat diperoleh model menyerupai *background* sebenarnya [19].

Langkah-langkah GMM adalah sebagai berikut:

1. Pilih area ROI dari *frame*
2. Kemudian dibentuk model-model GMM. Model-model GMM terbentuk dari data warna pixel berdasarkan waktu. Hasil model tersebut akan menjadi 2 bagian, model yang mencerminkan *background* dan model non-*background*.
3. Lakukan pencocokan setiap *frame* dengan *frame* model GMM yang telah dibentuk sebelumnya.
4. Di cek apakah model sudah sesuai atau tidak, jika tidak maka harus membuat model baru dan kembali ke langkah ketiga, apabila sudah sesuai maka akan dilanjutkan ke langkah selanjutnya.
5. Update parameter GMM kemudian normalisasi bobotnya.
6. Mengurutkan mode berdasarkan bobot, setelah itu barulah memilih GMM yang akan menjadi *background* dan *foreground*.

2.5 Metode Analisis *Blob*

Analisis *blob* ini menggunakan metode *connected component*, dimana di setiap kumpulan pixel yang tingkat keabuannya bernilai satu, dikategorikan sebagai satu objek. Setiap objek yang terdeteksi akan diberi label untuk mempermudah pengolahan.[15] Pada Gambar 2.3 menjelaskan tentang pemilihan area blob yang terdeteksi. Dari analisis *blob* tersebut akan diperoleh informasi tentang *centroid*, luas *area*, tinggi, dan lebar sebuah objek dari bentuk *rectangle*.



Gambar 2.3. Bagian a adalah representasi citra ukuran 10x10 *pixel* dan citra dengan nilai *pixel* 1 merupakan representasi dari objek (*foreground*). Sedangkan bagian b merupakan *blob* yang terdeteksi. [15]

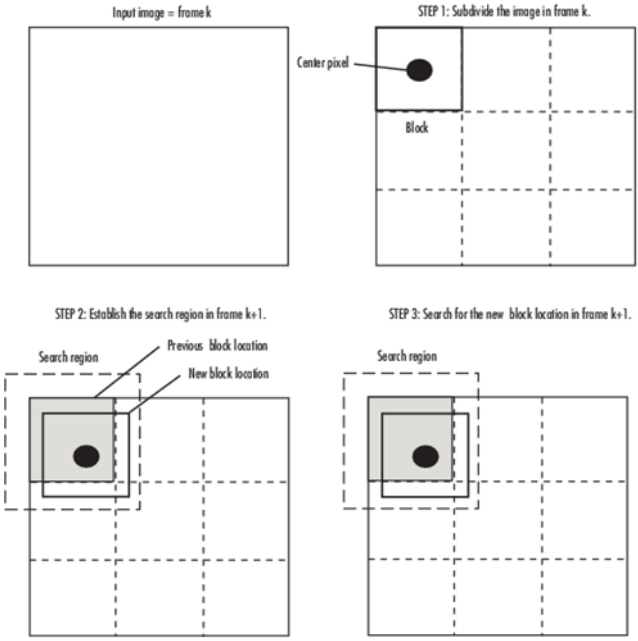
Berdasarkan Gambar 2.3 terlihat bahwa pemilihan *area blob* yang terdeteksi bergantung pada nilai *pixel*, jika berada dalam *area* yang memiki nilai *pixel* 1, maka *area* yang berada didalamnya akan terdeteksi, jadi *pixel* bernilai 1 sebagai batas untuk membedakan antara *background* dengan *foreground*.

2.6 Algoritma *Block Matching*

Block matching merupakan salah satu algoritma *motion estimation*. Tujuan dari teknik *motion estimation* adalah untuk mendapatkan vektor gerakan (displacement atau *velocity*) untuk setiap pixel pada citra. Algoritma ini mengestimasi gerakan pada blok ke blok. Area gerakan diasumsikan konstan dengan area blok persegi dan diwakili oleh vektor gerakan tunggal pada setiap bloknya [20]. Setiap blok pada *frame* sekarang dicocokkan dengan blok pada *frame* sebelumnya berdasarkan fungsi biaya tertentu. Beberapa fungsi biaya yang paling populer dan tidak mahal secara komputasi adalah Mean Absolute Difference (MAD) seperti pada Persamaan 2.1.

$$MAD = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left| C_{ij} - R_{ij} \right| \qquad (2.1)$$

dengan keterangan:
N = ukuran makro blok
C_{ij}= *pixel* pada makro blok sekarang
R_{ij}= *pixel* pada makro blok yang menjadi acuan



Gambar 2.4. Langkah-langkah dalam algoritma *block matching.* [17]

Gambar 2.3 menunjukkan langkah-langkah dalam algoritma *block matching*. Langkah-langkahnya adalah sebagai berikut.

1. Misalkan input ke blok adalah *frame* k, maka blok membagi *frame* dengan menggunakan nilai yang telah disesuaikan sebelumnya untuk ukuran blok.
2. Untuk setiap sub bagian atau sub blok dari *frame* k+1, pencocokan blok membentuk wilayah pencarian.
3. Blok akan mencari lokasi baru dari pergerakan berdasarkan wilayah pencariannya.

2.7 Algoritma *Particle Filter*

Particle Filter adalah metode state space untuk menerapkan filter Bayesian. Gagasan utamanya adalah mendekati perkiraan distribusi probabilitas *posterior* oleh partikel. Setiap partikel mewakili satu keadaan hipotetis objek, dengan sampling diskrit yang sesuai dengan probabilitas (berat). Partikelnya biasanya dilakukan resampling untuk meringankan degenerasi partikel. Efisiensi dan keakuratan filter partikel untuk pelacakan bergantung pada distribusi dan model observasi yang efektif untuk pembobotan partikel [11].

Dalam Bayesian, distribusi posterior dari state dapat ditulis sebagai berikut.

$$p(x_t/\bar{y}_t) = \beta\, p(y_t/x_t)\, p(x_t/\bar{y}_{t-1}) \qquad (2.2)$$

dengan β adalah faktor normalisasi, x_t adalah sistem state pada waktu ke- t, dan $\bar{y}_t$ adalah informasi yang terkumpul hingga waktu ke- t. Persamaan 2.2 berarti bahwa x_t bergantung pada observasi $\bar{y}_t$.

Particle Filter memberikan estimasi dari peluang posterior dari Persamaan 2.2 dalam 3 langkah, yaitu proses *sampling*, pembobotan dan *resampling*. Proses *sampling* terdiri dari mengambil *sample* (partikel) dari $p(x_t/\bar{y}_{t-1})$ yang disebut sebagai distribusi prior. Setelah itu dilakukan proses pembobotan, dimana hasil dari partikel tersebut diberi bobot dengan *likelihood* $p(y_t/x_t)$. Setelah dilakukan pembobotan maka akan dilakukan

resampling untuk menghindari degenerasi dari partikel yang tidak terpakai [21].

Dari Bayesian tersebut didapatkan pengembangannya yang salah satunya disebut *Particle Filter*. Algoritma *Particle Filter* yang digunakan dalam Tugas Akhir ini adalah *Adaptive Particle Filter*. *Adaptive Particle Filter* merupakan metode *Particle Filter* dengan menerapkan *adaptive motion model* untuk mendapatkan pendekatan distribusi yang lebih baik. Untuk lebih menyaring gangguan-gangguan yang ada, *motion continuity* dan kehalusan lintasan dikombinasikan dengan template korelasi dalam observasi *likelihood* [11].

Didefinisikan vektor keadaan objek sebagai $X = (x, y)$ dengan (x, y) adalah pusat objek. Model ruang keadaan dari objek yang dilacak adalah:

$$X_{t+1} = f(X_t, \mu_t) \quad (2.3)$$

$$Z_t = g(X_t, \xi_t) \quad (2.4)$$

dengan keterangan:

X_t = representasi vektor keadaan objek

$X_t = \begin{bmatrix} x \\ y \end{bmatrix}$ dimana x dan y adalah pusat objek

Z_t = vektor observasi, juga berbentuk $\begin{bmatrix} x \\ y \end{bmatrix}$

t = waktu

f = model dinamik

g = model observasi

μ_t = proses *noise*, digunakan bila ada *noise*

ξ_t = observasi *noise*, , digunakan bila ada *noise*

Dalam metode *Adaptive Particle Filter* ini dibedakan menjadi 2 yaitu model dinamik serta model observasi, yang masing-masing memiliki fungsi yang berbeda-beda. Model dinamik untuk menentukan peubahan keadaan objek pada frame,

sedangkan model observasi digunakan untuk melakukan pembobotan partikel berdasarkan fungsi likelihoodnya. [11]

- Model Dinamik

Model dinamik mencirikan perubahan keadaan objek pada *frame*. Estimasi gerak objek dinotasikan V_t dan μ_t adalah *error* prediksi keadaan dan dapat diperoleh dari $\left|uI_x + vI_y + I_t\right|$, u adalah besar gerakan pada sumbu x, v adalah besar gerakan pada sumbu y, sedangkan I_x , I_y dan I_t adalah turunan parsial dari fungsi intensitas terhadap x, y, *t*. Maka model dinamik (2.2) dapat dibentuk kembali menjadi :

$$X_{t+1} = X_t + V_t + \mu_t \qquad (2.5)$$

- Model observasi

Model observasi adalah model yang digunakan untuk melakukan pembobotan partikel berdasarkan fungsi *likelihood.* [11] Fungsi *likelihood* didefinisikan sebagai berikut :

$$P(Z_t|X_t) = P\left(Z_t^{int}\middle|\, X_t\right) P(Z_t^{mot}|\, X_t)^{O_{t-1}}\, P\left(Z_t^{trj}\middle|X_t\right)^{1-O_{t-1}} \qquad (2.6)$$

dengan $Z_t = \left\{Z_t^{int}, Z_t^{mot}, Z_t^{trj}\right\}$

$Z_t^{int} =$ pengukuran intensitas yang independen terhadap Z_t^{mot} dan Z_t^{trj}

$Z_t^{mot} =$ pengukuran gerakan

$Z_t^{trj} =$ pengukuran lintasan

$O_t =$ 0 jika objek mengalami oklusi dan 1 untuk yang lain.

Sebuah pengukuran intensitas dihitung berdasarkan kemiripan objek yang dilacak dengan partikel menggunakan *Sum of Square Differences (SSD)* yang didefinisikan sebagai berikut :

$$r(X_t) = \sum_{\chi \in W} [T(\chi) - I(\chi + X_t)]^2 \ , X_t \in Neib \qquad (2.7)$$

dengan keterangan:

W = *pixel* yang berada dalam ROI

$Neib$ = persekitaran yang kecil disekitar X_t

$T(\chi)$ = template objek yang dijadian acuan

I = *frame* pada waktu ke t.

Dari 2.6 diperoleh J kandidat yang merupakan kandidat yang cocok di dalam *Neib* tersebut sehingga *likelihood* intensitas diperoleh sebagai berikut :

$$P\left(Z_t^{int}|X_t\right) = q_0 U(.) + C_N \sum_{j=1}^{J} q_j N(r_t, \sigma_t) \qquad (2.8)$$

dengan keterangan:

$U.$ = sebuah kekacauan background yang diasumsikan berdistribusi uniform

C_N = faktor normalisasi

$q_j = \frac{1-q_0}{J}$ adalah peluang prior dengan $q_0 = 0.5$

Likelihood gerakan objek dihitung berdasarkan selisih antara perubahan posisi partikel dengan kecepatan rata-rata objek pada waktu sebelumnya yaitu :

$$d_{mot}^2 = (|\Delta x_t| - \overline{\Delta x})^2 + (|\Delta y_t| - \overline{\Delta y})^2 \ , \quad t > 1 \qquad (2.9)$$

dengan keterangan:

$(\Delta x, \Delta y)$ = perubahan posisi partikel

$(\overline{\Delta x}, \overline{\Delta y})$ = kecepatan rata-rata objek pada waktu sebelumnya.

Jika diketahui persamaan sebagai berikut:

$$\overline{\Delta x} = \sum_{s=t-k}^{t-1} \frac{|x_s - x_{s-1}|}{k} \ , \overline{\Delta y} = \sum_{s=t-k}^{t-1} \frac{|y_s - y_{s-1}|}{k} \qquad (2.10)$$

maka diperoleh perhitungan *likelihood* gerakan sebagai berikut :

$$P(Z_t^{mot}|X_t) = \frac{1}{\sqrt{2\pi}\sigma_{mot}} \exp\left(-\frac{d_{mot}^2}{2\sigma_{mot}^2}\right) \qquad (2.11)$$

dimana σ_{mot}^2 adalah varians untuk *likelihood* gerakan.

Sedangkan untuk *likelihood* lintasan diestimasi dari kedekatan partikel terhadap lintasan yang diperoleh dari posisi objek pada waktu sebelumnya. Didefinisikan fungsi lintasan dalam bentuk polinomial sebagai berikut:

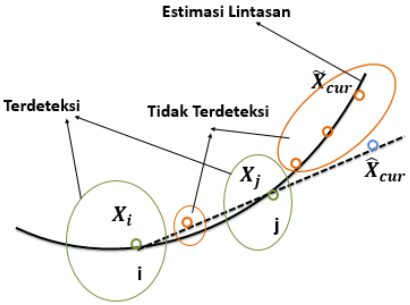
$$y = \sum_{i=0}^{m} a_i x^i \qquad (2.12)$$

dengan keterangan:
y = fungsi lintasan dalam bentuk polinomial
a_i = koefisien polinomial
m = order dari polinomial

Likelihood lintasan ini berdistribusi normal sehingga dapat ditulis sebagai berikut:

$$P\left(Z_t^{trj}\middle|X_t\right) = \frac{1}{\sqrt{2\pi}\sigma_{trj}} \exp\left(-\frac{d_{trj}/F^2}{2\sigma_{trj}^2}\right) \qquad (2.13)$$

dengan keterangan
σ_{trj}^2 = varians untuk *likelihood* lintasan,
$d_{trj} = \left|y - \sum_{i=0}^{m} a_i x^i\right|$
d_{trj} = *closeness metric*, adalah nilai kedekatan dari trayektori/ lintasan
$F = \lambda_f^{t\_o}$ adalah *forgotten factor* dan
λ_f = rasio *forgotten* $(0 < \lambda_f < 1)$
t_o = banyaknya *frame* pada saat objek mengalami oklusi.
Gambar 2.5 menunjukkan ilustrasi pencocokan lintasan objek (warna kuning).



Gambar 2.5. Ilustrasi pencocokan lintasan objek. [15]

- Penghalusan posisi

Jika objek terdeteksi pada *frame* sebelumnya atau *cur-1* $(O_{cur-1} = 1)$, estimasi keadaan dilakukan dengan menghitung rata-rata bobot dari semua partikel, namun jika tidak terdeteksi

pilih satu atau lebih partikel dengan bobot maksimum dan hitung rata-rata partikel tersebut.

Penghalusan posisi objek dengan cara memproyeksikan salah satu hasil prediksi pada estimasi lintasan hanya dilakukan jika $O_{cur} = 0$. Diberikan dua posisi saat benda terdeteksi yaitu X_j dan X_i pada *frame* j dan i , $(j > i)$ prediksi posisi $\hat{X}_{cur}$ dihitung sebagai :

$$\hat{X}_{cur} = X_j + \frac{(X_j - X_i)(cur - j)}{j - i} \qquad (2.14).$$

Proyeksi dari $\hat{X}_{cur}$ yaitu $\tilde{X}_{cur}$ didefinisikan sebagai titik pada lintasan yang terdekat dengan $\hat{X}_{cur}$. Sehingga penghalusan posisi objek adalah sebagai berikut :

$$X_{cur} = \left(1 - \lambda_f^{t\_o}\right)\hat{X}_{cur} + \tilde{X}_{cur}\,\lambda_f^{t\_o} \qquad (2.15).$$

Setelah mendapatkan posisi benda pada *frame* tersebut kemudian proses pelacakan dilanjutkan pada *frame* berikutnya. Dari beberapa penjelasan diatas, prosedur secara umum tentang algoritma *Adaptive Particle Filter* dapat ditunjukkan sebagai berikut.

Algoritma *Adaptive Particle Filter*

Partikel pada waktu $\boldsymbol{t}$ yaitu $\left\{\left(\boldsymbol{X_{t-1}^{(i)}}, \boldsymbol{\pi_{t-1}^{(i)}}\right) \middle| \boldsymbol{i = 1, \ldots, N}\right\}$, untuk waktu ke $\boldsymbol{t}$ lakukan langkah berikut :

Prediksi

if $\boldsymbol{O_{t-1} = 1}$

do estimasi gerak $\boldsymbol{V_t}$ dan prediksi error $\boldsymbol{\mu_t}$

else

$\boldsymbol{V_t = 0, \mu_t = \mu_{max}}$

endif

for $\boldsymbol{i = 1}$: $\boldsymbol{N}$

$\boldsymbol{X_t^{(i)} \sim N\left(X_{t-1}^{(i)} + V_t, \mu_t\right)}$

Endfor

Update
for $\boldsymbol{i = 1}$: $\boldsymbol{N}$
$\boldsymbol{\pi_t^{(i)} = P\left(Z_t \middle| X_t^{(i)}\right)}$ sesuai dengan persamaan (2.6).
Endfor

Resampling (jika diperlukan)
Ganti $\left\{\left(\boldsymbol{X_t^{(i)}, \pi_t^{(i)}}\right) \middle| \boldsymbol{i = 1, \dots, N}\right\}$ dengan
$\left\{\left(\boldsymbol{\tilde{X}_t^{(i)}, \frac{1}{N}}\right) \middle| \boldsymbol{i = 1, \dots, N}\right\}$

Estimasi
if $\boldsymbol{O_{t-1} = 1}$
Hitung rata-rata dari semua partikel berdasarkan setiap bobot partikel.
else
pilih satu atau lebih partikel dengan bobot maksimum dan hitung rata-rata partikel tersebut. Kemudian lakukan penghalusan posisi menggunakan persamaan (2.15).
end

BAB III
METODOLOGI

Tahapan pengerjaan dalam menyelesaikan Tugas Akhir ini terdiri dari 6 tahapan, yaitu studi literatur dan perekaman video, analisis dan desain sistem, implementasi sistem, uji coba dan pembahasan, penarikan kesimpulan, serta penyusunan laporan tugas akhir.

3.1 Studi Literatur dan Perekaman Video

Pada tahap pertama ini akan dilakukan pengkajian tentang algoritma pelacakan serta penghitungan orang. Pada tahap ini juga akan dilakukan perekaman video yang berisi orang yang berlalu-lalang. Studi ini dilakukan dengan membaca buku, jurnal, ataupun artikel yang terkait serta melakukan pengambilan video.

Video diambil menggunakan kamera digital, yang dimana dalam pengambilan video tersebut dilakukan beberapa pengukuran parameter. Rekaman video diambil berdasarkan ketentuan berikut:

a. Video didapat dari lingkungan outdoor.
b. Video didapat dengan menggunakan kamera digital.
c. Video beresolusi videonya adalah 640x480.

3.2 Analisis dan Desain Sistem

Pada tahap kedua ini akan dilakukan analisis video dan penentuan parameter-parameter apa yang akan dibutuhkan dalam pembuatan program. Kemudian dibuat desain sistem dari program sesuai dengan hasil analisis. Berikut beberapa tahapan untuk mengolah rekaman video *offline*:

a. Sistem ini memiliki inputan berupa video digital. Pemilihan area ROI (*Region of Interest*) pada rekaman video yang digunakan sebagai area pendeteksian.

b. *Background Subtraction* menggunakan metode GMM untuk segmentasi setiap *frame* menjadi citra *background* dan *foreground*.
c. Analisis *blob* pada citra foreground untuk mendeteksi objek pada citra. Selanjutnya proses *tracking* untuk menentukan apakah objek pada *frame* ke-i dengan *frame* ke- i-1 merupakan objek yang sama.
d. Objek-objek itu akan dicari *centroid*nya yang nantinya akan digunakan untuk mendeteksi orang dengan menggunakan metode *particle filter*, kemudian dilanjutkan dengan penghitungan orang yang telah terdeteksi.

Untuk mengimplementasikan beberapa kebutuhan yang telah dicatat pada tahap analisis. Perangkat lunak pemrograman yang dipakai untuk membuat program ini adalah MATLAB.

3.3 Implementasi Sistem

Pada tahap ketiga akan dilanjutkan dengan implementasi sistem dalam bentuk perangkat lunak sesuai dengan hasil analisis dan desain sistem. Akan dibuat desain *interface* yang menarik dan *user friendly* untuk memudahkan dan membuat nyaman pengguna.

3.4 Uji Coba dan Pembahasan

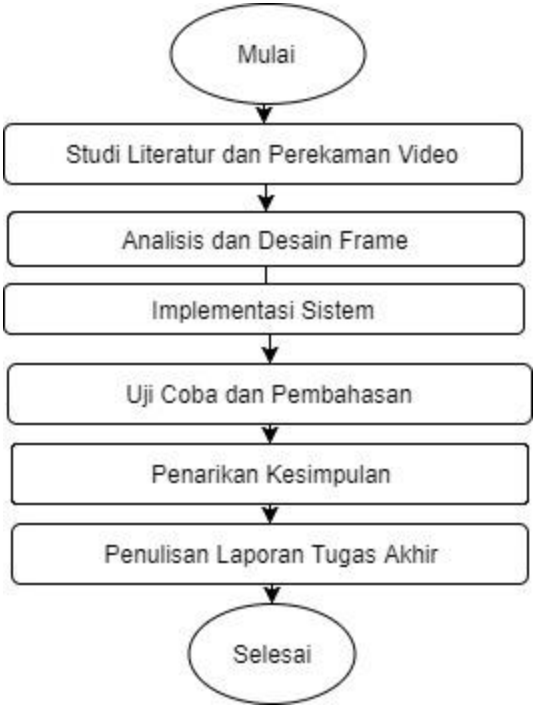
Pada tahap ini dilakukan pengujian terhadap perangkat lunak yang telah selesai dibuat menggunakan input video rekaman orang yang berlalu-lalang. Kemudian dilakukan analisis pembahasan sehingga dapat dicatat beberapa hal yang dijadikan pertimbangan dalam menarik kesimpulan. Pada tahap ini juga dilakukan evaluasi terhadap program yang telah dibangun. Hasil evaluasi dicatat untuk membenahi hal-hal yang masih kurang. Kemudian dilanjutkan dengan uji coba deteksi serta perhitungan orang yang berada dalam area ROI serta dicari tingkat akurasinya. Selain itu akan dilakukan evaluasi sehingga dapat membenahi hal-hal yang masih kurang dan mengatasi *error* yang ditemukan.

3.5 Penarikan Kesimpulan

Tahap kelima ini dilakukan penarikan kesimpulan dari hasil pengerjaan dan memberikan saran untuk pengembangan berikutnya.

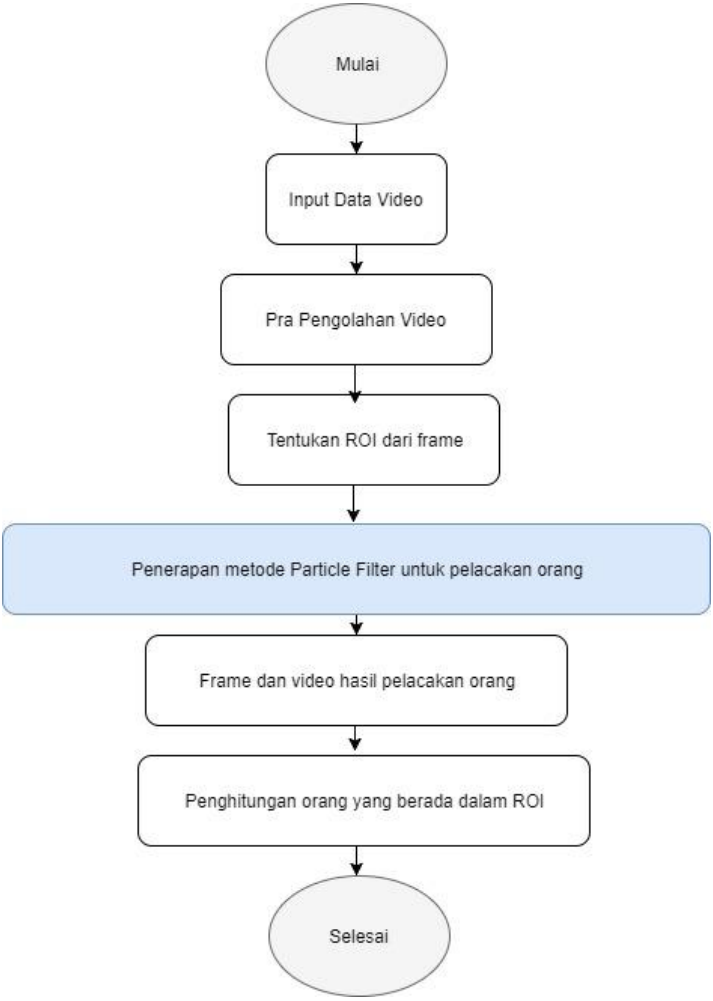
3.6 Penulisan Laporan Tugas Akhir

Pada tahap terakhir ini penulis menuliskan semua hasil yang telah didapatkan selama pengerjaan Tugas Akhir. Adapun blok diagram dari proses pengerjaan Tugas Akhir ini dapat dilihat pada Gambar 3.1 berikut.



Gambar 3.1. Blok Diagram Proses Pengerjaan

Sedangkan diagram alur program Aplikasi Pelacakan Serta Penghitungan Orang Menggunakan Metode *Particle Filter* dapat dilihat pada Gambar 3.2. berikut.



Gambar 3.2. Diagram alur program Aplikasi Pelacakan Serta Penghitungan Orang Menggunakan Metode *Particle Filte*

BAB IV
PERANCANGAN DAN IMPLEMENTASI

Pada bab ini akan dibahas mengenai perancangan dan implementasi prototipe perangkat lunak dimulai dari pembahasan proses pengambilan data masukan, pengolahan data masukan dengan metode *Particle Filter* untuk proses pelacakan orang, serta penjelasan mengenai cara untuk mendapatkan data keluaran yang sesuai dengan tujuan dari penelitian Tugas Akhir ini.

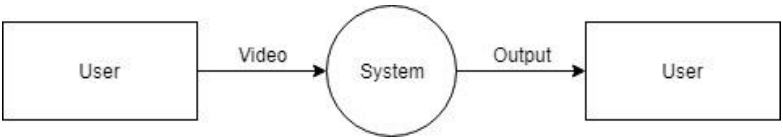
4.1 Analisis Prototipe Perangkat Lunak

Untuk mengetahui gambaran secara keseluruhan dari proses kerja prototipe perangkat lunak yang akan dibuat, maka diperlukan langkah awal dalam pembuatan prototipe perangkat lunak yaitu melakukan analisis kerja prototipe perangkat lunak secara keseluruhan. Prototipe perangkat lunak yang dibuat digunakan untuk melakukan uji coba pelacakan orang.

Program Pelacakan Orang menggunakan metode *Particle Filter* merupakan program utama dalam prototipe perangkat lunak ini. Fungsinya adalah melakukan estimasi gerakan objek dengan metode *Hierarchical Block Matching*. Hasil estimasi gerakan tersebut ditambahkan pada model yang digunakan untuk melakukan pelacakan objek dengan metode *Particle Filter*, sehingga dihasilkan suatu output video pelacakan objek. Berikut ini akan disajikan *Data Flow Diagram* dari program tersebut.

4.1.1 *Data Flow Diagram* Level 0

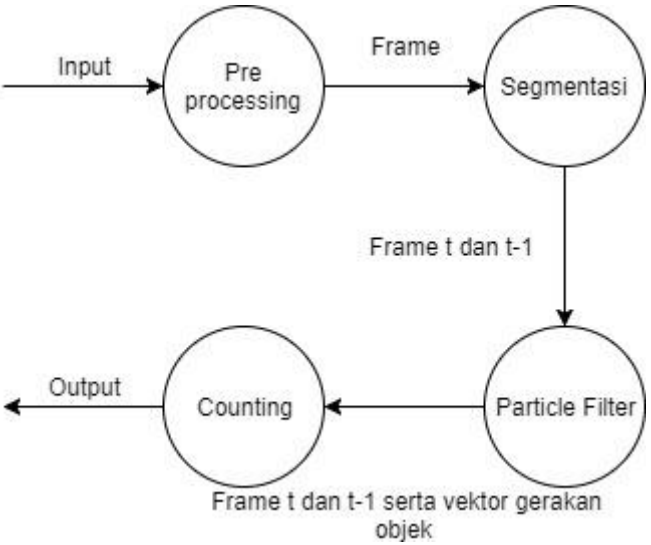
Berikut ini adalah *data flow diagram* level 0 dari program tersebut



Gambar 4.1. *Data Flow Diagram* Level 0.

4.1.2 *Data Flow Diagram* Level 1

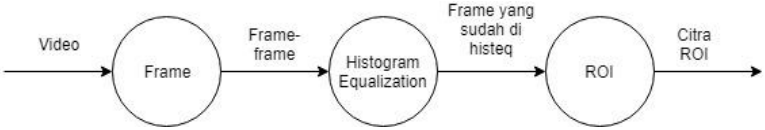
Berikut ini adalah *data flow diagram* level 1 dari program tersebut



Gambar 4.2. *Data Flow Diagram* Level 1.

4.1.3 *Data Flow Diagram* Level 2 (*Preprocessing*)

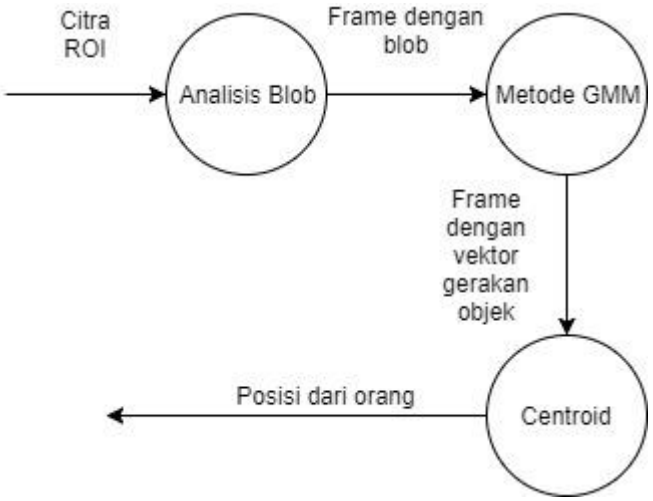
Berikut ini adalah *data flow diagram* level 2 untuk proses *pre-processing* dari program tersebut.



Gambar 4.3. *Data Flow Diagram* Level 2 untuk proses *preprocessing.*

4.1.4 *Data Flow Diagram* Level 2 (Segmentasi)

Berikut ini adalah *data flow diagram* level 2 untuk proses segmentasi dari program tersebut.



Gambar 4.4. Data Flow Diagram Level 2 untuk proses segmentasi.

4.2 Perancangan Prototipe Perangkat Lunak

Setelah analisis perangkat lunak, kemudian dilanjutkan dengan perancangan perangkat lunak. Perancangan perangkat lunak tersebut meliputi lingkungan perancangan, perancangan data perangkat lunak dan perancangan antarmuka.

4.2.1 Lingkungan Perancangan Perangkat Lunak

Lingkungan perancangan perangkat lunak dalam Tugas Akhir ini meliputi perangkat keras dan perangkat lunak yang digunakan. Spesifikasinya disajikan dalam Tabel 4.1.

Tabel 4.1. Lingkungan Perancangan Prototipe Perangkat Lunak

Perangkat keras	1. Intel Core i5 processor 2400S 2.5GHz 2. Memory 4096MB RAM
Perangkat lunak	1. Sistem operasi Microsoft Windows 7 Pro 64 bit 2. Tools menggunakan MATLAB R2017a.

4.2.2 Perancangan Data Perangkat Lunak

Terdapat tiga macam data yang digunakan oleh perangkat lunak ini antara lain data masukan, data proses, dan data keluaran. Pada Tugas Akhir ini data masukan berupa video orang yang berjalan . Data proses merupakan data yang berisi parameter-parameter yang akan digunakan oleh Hierarchical Block Matching serta algoritma pelacakan orang dengan metode Particle Filter. Sedangkan data keluaran adalah data hasil pelacakan orang.

Data masukan sistem ini berupa video orang yang berjalan. File video yang digunakan memiliki spesifikasi sesuai dengan batasan masalah yang terdapat dalam penelitian Tugas Akhir.

Data proses merupakan data yang digunakan dalam proses pengolahan data masukan. Data proses ini diperoleh dari hasil pengolahan data masukan sesuai dengan tahapan algoritma dan metode yang telah disusun. Tabel 4.2 menjelaskan tahapan dari data proses.

Tabel 4.2. Tabel Data Proses

No	**Tahapan**	***Input***	***Output***
1.	*Input* Awal	Video	*Frame* Citra

2	Pra pengolahan video	Video	*Frame* Citra dengan histogram yang lebih merata
3.	Pilih Area ROI	*Frame* pertama dari video	Citra ROI
4.	Proses *Hierarchical Block Matching*	*Frame* t dan $t-1$	Vektor gerakan orang
5.	Proses Pelacakan Orang dengan metode *Particle Filter*	*Frame* Citra pada $t-1$ dan t serta vektor gerakan objek	Posisi baru dari orang
6	Penghitungan orang yang berada dalam area	*Frame* Citra pada $t-1$ dan t serta vektor gerakan objek	Jumlah orang yang berada dalam area

Data keluaran yang dihasilkan dari prototipe perangkat lunak ini adalah berupa semua frame dan video hasil pelacakan orang dengan penandaan posisi target (orang) di setiap frame.

4.3 Implementasi Sistem

Implementasi prosedur pelacakan orang menggunakan metode *Particle Filter* terdiri dari beberapa tahapan meliputi pengambilan input video, pra pengolahan video, pemilihan area ROI, pelacakan orang dengan metode *Particle Filter* dan estimasi gerakan dengan *Hierarchical Block Matching*.

4.3.1 Implementasi Input Video

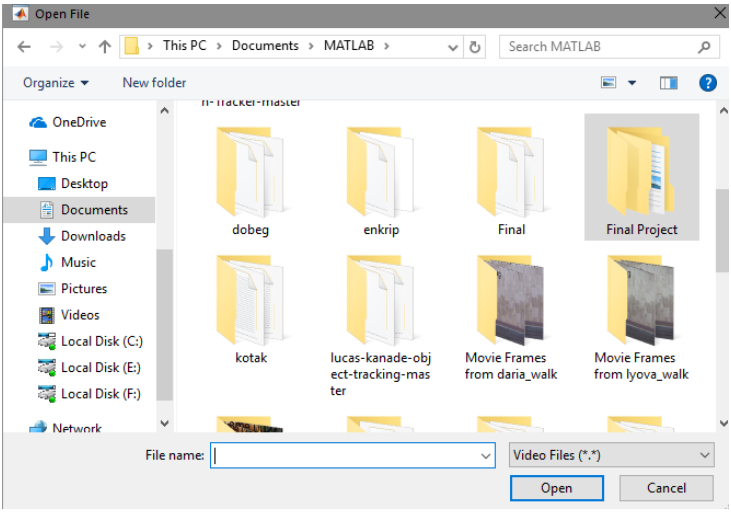
Pada proses ini akan ditentukan video yang nantinya digunakan sebagai data masukan untuk diproses. Penjabaran tentang proses pemilihan video adalah sebagai berikut**.**

Fungsi : menginputkan video bertipe *.avi pada form simulasi.
Input : video bertipe *.avi
Output : video bertipe *.avi
Deskripsi : mengambil video bertipe *.avi, *.mp4. *.mpeg yang telah disimpan pada komputer.
Gambar 4.3 adalah tampilan antar muka pengambilan *input* video. Kode program untuk menerima *input* video adalah sebagai berikut :

```
global vid reader backg
imaqreset
set(hObject,'UserData',0)
set(handles.framech,'Enable','off');
set(handles.reset,'Enable','off');
set(hObject,'Enable','off');
 [filename, pathname]=uigetfile('*.avi','SELECT
VIDEO AVI');
        if filename == 0
set(hObject,'Enable','on');
            set(handles.framech,'Enable','on');
            set(handles.reset,'Enable','on');
            return
        end

        vid = fullfile(pathname,filename);
        set(handles.vidname,'String',filename);
        reader = vision.VideoFileReader(vid);
        img = step(reader);
        axes(handles.axes1);
        imshow(img);
        axis image off
        handles.img     =img;
        handles.pathname=pathname;
```

```
        handles.filename=filename;
guidata(hObject,handles)
set(hObject,'Enable','on');
set(handles.framech,'Enable','on');
set(handles.reset,'Enable','on');
set(handles.tracking,'Enable','on');
```



Gambar 4.5. Antar muka input video

4.3.2 Implementasi Pra Pengolahan Video

Pada proses ini akan dilakukan perubahan dari video menjadi frame yang akan diubah dengan ekualisasi histogram. Penjabaran prosesnya adalah sebagai berikut.

```
global reader
myFolder='D:\TA Faul\data';
imga=0001;
while ~isDone(reader)
frame=step(reader);
as=rgb2gray(frame);
```

```
go=histeq(as);
filename=strcat('Frame',num2str(imga),'.png');

fullname=fullfile(myFolder,filename);
imwrite(go,fullname);
imga=imga+1;
end
```

4.3.3 Implementasi Pemilihan ROI

Region of Interest (ROI) adalah daerah dari citra atau *frame* yang didefinisikan sebagi objek yang akan dilacak. Penjabaran tentang proses pemilihan ROI objek adalah sebagai berikut.

Fungsi : menentukan objek yang akan dilacak pergerakannya di setiap frame.
Input : frame pertama dari video.
Output : citra ROI.
Deskripsi : mendapatkan citra dari objek yang akan dilacak.
Pembentukan ROI objek diperoleh dengan membentuk *rectangle*. Kode Program untuk pemilihan ROI objek adalah sebagai berikut :

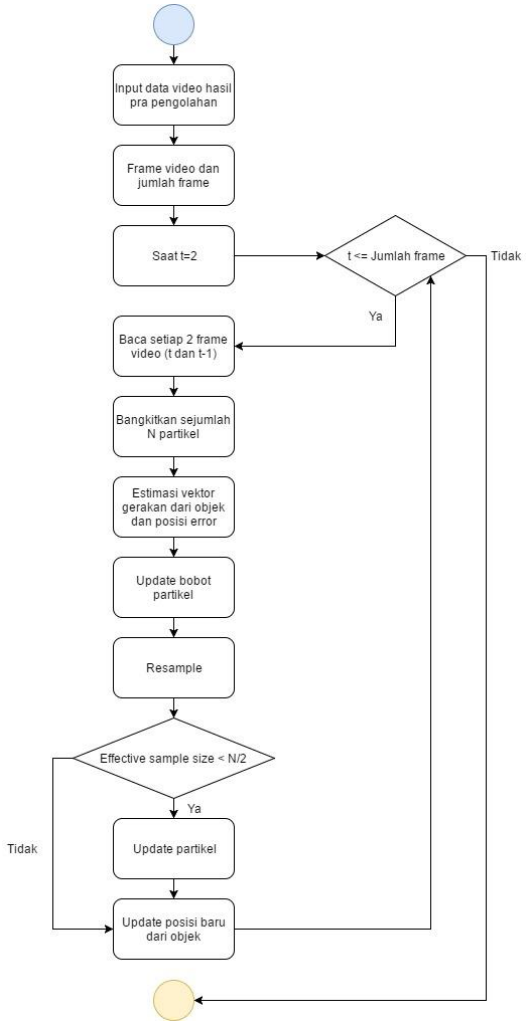
```
global position a b c d e f roi h
image = handles.img ;
h = imrect;
setColor(h,'red');
position = wait(h);
roi = getPosition(h)
```

4.3.4 Implementasi Pelacakan Orang dengan Metode *Particle Filter*.

Pada proses ini akan diproses pelacakan orang dengan metode *Particle Filter*. Untuk source code proses pelacakan objek bergerak dengan metode *Particle Filter* dapat dilihat pada lampiran. Penjabaran tentang proses metode *Particle Filter* adalah sebagai berikut.

Fungsi	: memproses setiap frame untuk dilakukan pelacakan orang yang telah ditentukan.
Input	: frame video pada t-1 dan t serta vektor gerakan objek.
Output	: posisi baru dari objek.
Deskripsi	: memproses setiap frame video yang untuk selanjutnya dilakukan pelacakan objek.

Berikut ini adalah diagram alur penerapan metode *Particle Filter* untuk pelacakan orang.



Gambar 4.6. Diagram alur penerapan metode *Particle Filter* untuk pelacakan orang.

Penjelasan mengenai setiap tahapan dalam proses pelacakan orang menggunakan metode *Particle Filter* adalah sebagai berikut:

1. Pembacaan setiap 2 frame berurutan

Pembacaan dua frame berurutan bertujuan untuk melakukan estimasi gerakan dengan menggunakan metode *Hierarchical Block Matching*. Frame yang digunakan adalah frame ke t dan $t-1$. Metode diterapkan pada saat $t=2$ karena pada saat $t=1$ dilakukan inisialisasi objek yang akan dilacak. Untuk frame selanjutnya dilakukan pembacaan dua frame berurutan sampai dengan jumlah frame video. Proses pembacaan citra diimplementasikan dalam sebuah fungsi sebagai berikut :

```
function citra = readFrame()

      citra = step(obj.videoReader);

end
```

2. *Bangkitkan sejumlah N partikel*

Pada kondisi awal, partikel $X_t^{(i)}$disebarkan secara acak dengan informasi mengenai bobot partikel yang harus diberikan belum diketahui. Maka digunakan prinsip *noninformative prior* yang menyatakan bahwa jika tidak ada yang diketahui mengenai x, maka prior $p(x)$ dinyatakan mempunyai distribusi uniform, yaitu seluruh *outcome* dari x yang mungkin mempunyai probabilitas yang sama. Jadi, berdasarkan teori di atas, pembobotan awal partikel dilakukan dengan menggunakan distribusi uniform, bobot awal partikel $\pi_t^{(i)} = 1/N$ dimana *N* adalah jumlah partikel yang digunakan. Kode program untuk membangkitkan sejumlah N partikel adalah sebagai berikut :

```
X = bsxfun(@plus,trackedLocation(t-
1,:)',sqrt(varians)*randn(2,N));
```

3. Estimasi vektor gerakan dari objek dan prediksi error.

Vektor gerakan objek diperoleh dengan menggunakan metode *Hierarchical Block Matching*. Penjelasan mengenai estimasi gerak terdapat pada subbab 4.4.5. Sedangkan untuk prediksi error diperoleh menggunakan Persamaan 2.6. Suatu citra digital pada video saat frame t dapat dinyatakan dengan fungsi $f(x, y, t)$, dimana *x* dan *y* merupakan titik koordinat spasial dan t adalah urutan frame. Turunan parsial dari $f(x, y, t)$ terhadap x, y, t adalah

$$\mathrm{I_x} = \frac{\partial \mathrm{f}}{\partial \mathrm{x}} = \mathrm{f(x+1, y, t)} - \mathrm{f(x, y, t)} \quad (4.1)$$

$$\mathrm{I_y} = \frac{\partial \mathrm{f}}{\partial \mathrm{y}} = \mathrm{f(x, y+1, t)} - \mathrm{f(x, y, t)} \quad (4.2)$$

$$\mathrm{I_t} = \frac{\partial \mathrm{f}}{\partial \mathrm{t}} = \mathrm{f(x, y, t+1)} - \mathrm{f(x, y, t)} \quad (4.3)$$

Kode program untuk menghitung prediksi error adalah sebagai berikut:

```
pixel = round(trackedLocation(t-1,:));
Ix = prevFrame(pixel(2),pixel(1)+1)-
prevFrame(pixel(2),pixel(1));
Iy = prevFrame(pixel(2)+1,pixel(1))-
prevFrame(pixel(2),pixel(1));
It = frame2(pixel(2),pixel(1)) -
prevFrame(pixel(2),pixel(1));
```

Setelah prediksi error diperoleh, partikel $\left\{\left(X_{t-1}^{(i)}, \pi_{t-1}^{(i)}\right) \middle| i = 1, \ldots, N\right\}$ *digantikan menjadi* $X_t^{(i)} \sim N(X_{t-1}^{(i)} + V_t, \mu_t)$ dengan V_t adalah vektor gerakan objek. Jika $V_t = 0$ maka $\mu_t = \mu_{maks}$.

4. *Update bobot partikel*

Perubahan bobot partikel dilakukan dengan menghitung pengukuran intensitas, pengukuran gerakan dan pengukuran lintasan. Pengukuran intensitas tidak bergantung pada pengukuran

gerakan dan pengukuran lintasan. Apabila terdeteksi adanya gerakan objek atau objek tidak mengalami oklusi maka bobot partikel diubah berdasarkan pengukuran intensitas dan pengukuran gerakan. Sedangkan bila tidak terdeteksi gerakan objek atau objek mengalami oklusi maka bobot partikel diubah berdasarkan pengukuran intensitas dan pengukuran lintasan.

Pengukuran intensitas diperoleh dengan menghitung kemiripan antara target dan kandidat partikel. Perhitungan kemiripan antara target dengan kandidat partikel diperoleh dengan menghitung SSD correlation surface. SSD correlation surface diimplementasikan dalam sebuah fungsi sebagai berikut

```
function [ J , stdIT , r ] = SSD( frame,
template, particle, N, neib)
```

Variabel frame adalah frame video pada saat t, variabel template adalah citra template objek, variabel particle adalah seluruh partikel, variabel N adalah jumlah partikel dan variabel neib adalah persekitaran partikel. Output dari fungsi tersebut adalah J yaitu banyaknya kandidat dalam neib untuk setiap partikel, stdIT yaitu standar deviasi dan r adalah nilai SSD untuk setiap partikel. Kode program selengkapnya dari fungsi untuk menghitung SSD setiap partikel disajikan pada Lampiran A1.

Setelah diperoleh SSD untuk setiap partikel, dilakukan perhitungan pengukuran intensitas seperti pada Persamaan 2.9, bab kajian teori. Kode program untuk perhitungan pengukuran intensitas adalah sebagai berikut:

```
[ J , stdIT , r ] = SSD (frame2,template, Xt ,N
, mbSize);

a = 0; b = 1;

q0 = 0.5;
C = 1/sum(r);
```

```
for i= 1:N

qj = (1-0.5)/J(i);

p = qj*(normrnd(r(i),stdIT(i),[1 J(i)]));

P1(i)= (q0*unifrnd(a,b))+(C*sum(p));end
```

Pengukuran gerakan dihitung berdasarkan selisih antara perubahan posisi partikel dan rata-rata perubahan posisi objek. Kode program untuk pengukuran gerak sebagai berikut:

```
%average object speed
delta2 = 0;
for o=-30:-1
if(t+o<=0)
    Xs = [0 0];
else
     Xs = trackedLocation(t+o,:);
end
if(t+o-1<=0)
  Xs1 = [0 0];
else
  Xs1 = trackedLocation(t+o-1,:);
End

delta2_0 = Xs - Xs1;
delta2 = delta2 + delta2_0;
end
delta2 = delta2/30 ;


%particle speed

for i=1:N

delta1 =[ abs(X(1,i)-Xt(1,i)) abs(X(2,i)-
Xt(2,i))];

d = ((delta1(1)- delta2(1))^2+ (delta1(2)-
delta2(2))^2);

P2(i)= (1/sqrt(2*pi))*(exp(-1/2 *(d/50).^2));

end
```

Pengukuran lintasan diestimasi dari kedekatan partikel dengan lintasan. Lintasan diperoleh dari posisi objek sebelumnya dan dinotasikan dalam fungsi polinomial dengan order 2. Pada pengukuran lintasan juga terdapat λ_f *yaitu rasio forgotten* $(0 < \lambda_f < 1)$ *dan t_o* adalah banyaknya frame pada saat objek tidak terdeteksi. Pada Tugas Akhir ini digunakan rasio forgotten 0,9. Kode program untuk pengukuran lintasan adalah sebagai berikut:

```
occ = occ+1;

koef_poly = polyfit
(trackedLocation(:,1),trackedLocation(:,2),2)
;

for i=1:N

poly = [ Xt(2,i)^2 Xt(2,i) 1];

d_trj = abs (Xt(2,i)-(sum(koef_poly.*poly)));

P2(i)= (1/sqrt(2*pi))*(exp(-1/2
*((d_trj/(0.9^occ))/1).^2));

End
```

5. *Resampling*

Proses *resampling* bertujuan untuk menghilangkan partikel dengan bobot rendah. *Resampling* dilakukan jika nilai *Effective Sample Size* kurang dari setengah jumlah partikel.

Proses *resampling* akan menggantikan *set* partikel $\left\{\left(X_t^{(i)}, \pi_t^{(i)}\right) \middle| i = 1, \dots, N\right\}$ menjadi $\left\{\left(\tilde{X}_t^{(i)}, \frac{1}{N}\right) \middle| i = 1, \dots, N\right\}$.

Proses *resampling* diimplementasikan dalam sebuah fungsi yaitu :

```
function [ indx ] = resample(w)
```

```
[index] = resample(w);

Xt = Xt(:,index);
```

Fungsi resample dengan input variabel w sebagai bobot partikel akan menghasilkan output berupa suatu indeks. Indeks tersebut digunakan untuk memilih partikel yang akan diestimasi. Kode program selengkapnya dari fungsi untuk proses *resampling* disajikan pada Lampiran A2.

6. *Update* posisi objek

Posisi terbaru dari objek diperoleh dengan cara mengestimasi posisi objek. Estimasi posisi dilakukan dengan menghitung nilai rata-rata dari nilai seluruh partikel berdasarkan setiap bobotnya. Kode program untuk estimasi posisi objek adalah sebagai berikut:

```
Detect (j,:) =
(sum(bsxfun(@times,Xt,w),2)/sum(w))';

trackedLocation(t,:) = Detect(j,:);
```

Apabila pada frame $t-1$ tidak terdeteksi gerakan atau terjadi oklusi maka estimasi dilakukan dengan cara memilih satu atau lebih partikel dengan bobot maksimum dan hitung rata-rata partikel tersebut untuk selanjutnya dilakukan penghalusan posisi. Penghalusan posisi objek diawali dengan menentukan satu prediksi posisi objek $\hat{X}_{cur}$. Kemudian mencari proyeksi dari $\hat{X}_{cur}$ yaitu $\tilde{X}_{cur}$ yang didefinisikan sebagai titik pada lintasan yang terdekat dengan $\hat{X}_{cur}$. Untuk menghitung titik terdekat dengan $\hat{X}_{cur}$ digunakan jarak *Euclidean*. Sehingga untuk posisi terbaru hasil penghalusan adalah $X_{cur} = \left(1-\lambda_f^{t\_o}\right)\hat{X}_{cur} + \tilde{X}_{cur}\lambda_f^{t\_o}$. Kode program untuk penghalusan posisi adalah sebagai berikut :

```
cur= Detect(j,:)+((Detect(j,:)-Detect(j-1,:))
* ((t-point(j,1))/(point(j,1)-point(j-1,1))))
point_trj = trackedLocation(1:end-1,:);
D = pdist2(point_trj,cur,'euclidean');
closest = find(D == min(D(:)));
closest_point = trackedLocation(closest(1),:);
trackedLocation(t,:) =((1-
0.9^occ)*cur)+(closest_point*(0.9^occ));
```

Proses pelacakan objek diimplementasikan dalam sebuah fungsi sebagai berikut :

```
function [trackedLocation] = adaptivePF (
video, roi, mbSize , search, partikel ,frame)
```

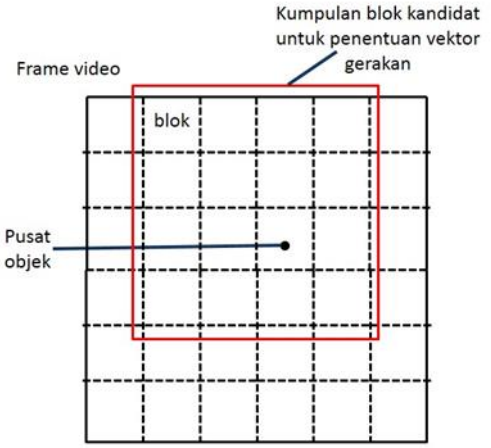
Variabel video adalah video input untuk pelacakan objek, sedangkan roi adalah koordinat [$x\ y\ width\ height$] dari objek. Variabel mbSize adalah ukuran blok untuk estimasi gerakan objek, sedangkan variabel search adalah besar area pencarian. Variabel partikel adalah jumlah partikel untuk penerapan metode *Adaptive Particle Filter* sedangkan variabel frame adalah seluruh frame hasil pra pengolahan. Kode program selengkapnya dari fungsi untuk proses pelacakan objek dengan metode *Particle Filter* disajikan pada Lampiran B2.

4.4 Proses Estimasi Gerakan Objek

Estimasi gerakan objek dilakukan dengan menggunakan metode Hierarchical Block Matching.

Fungsi : mendapatkan vektor gerakan objek.
Input : frame t dan t-1.
Output : vektor gerakan objek.
Deskripsi: estimasi gerakan yang dilakukan adalah berbasis blok. Besar blok telah ditentukan oleh user saat pengisian parameter. Begitu pula untuk besarnya area pencarian.

Proses estimasi gerakan objek dengan hanya memperhitungkan blok di sekitar objek. Kemudian memilih vektor gerakan yang terbesar yang terpilih sebagai gerakan objek. Gambar 4.4 mengilustrasikan proses estimasi gerakan objek.



Gambar 4.7 Ilustrasi estimasi gerakan objek [15]

Proses estimasi gerakan objek diimplementasikan ke dalam sebuah fungsi sebagai berikut :
Variabel Ref_Img adalah *frame* video pada waktu t-1 , sedangkan variabel Target_Img adalah *frame* video pada t. Variabel Msize adalah ukuran blok, sedangkan variabel sr adalah besar area pencarian. Variabel loct adalah posisi objek dalam koordinat (x,y) pada waktu t-1. Kode program selengkapnya dari fungsi untuk proses estimasi gerak menggunakan metode *Hierarchical Block Matching* disajikan pada Lampiran B1.

4.5 Proses Penghitungan Orang

Setelah didapatkan orang yang terdeteksi maka tinggal menghitung jumlah orang yang berada dalam area ROInya.

Fungsi : menghitung orang yang berada dalam area
Input : frame citra pada t-1 dan t serta vektor gerakan objek
Output : jumlah orang yang berada dalam area
Deskripsi : menghitung jumlah orang yang berada dalam area ROI

Proses ini dilakukan untuk menghitung orang yang berada dalam area ROI, penghitungan jumlah orang untuk mendeteksi orang tersebut, dengan kondisi dapat dihitung jika centroid dari orang tersebut memasuki area ROI. Proses penghitungan orang dapat diimplementasikan dalam fungsi berikut.

```
num=size(bbox,1);
set(handles.numpeo,'String',num);
axes(handles.axes1);
imshow(result);
drawnow
for n=1:num
    hold on
    centroid = s(n).Centroid;
    X = round(centroid(1));
    Y = round(centroid(2));
      if X>roi(1) && X<roi(1)+roi(3)
      if Y<roi(2) && Y>roi(2)+roi(4)
            set(handles.numpeo,'String',num);
      end
      end
      else
            set(handles.numpeo,'String',0);
      end
    hold off
```

BAB V
UJI COBA DAN PEMBAHASAN

Bab ini menjelaskan mengenai pengujian program dan pembahasan dari hasil uji coba. Pengujian yang dilakukan disini adalah pengujian program dengan *input* video orang yang sedang berlalu lalang. Video adalah video yang berada diluar serta dalam ruangan.

5.1 Data Uji Coba

Uji coba pada program dalam Tugas Akhir ini dilakukan terhadap video *.avi. Video uji coba sudah tersimpan dalam penyimpanan komputer dan diperoleh dari hasil rekaman. Video yang digunakan sebanyak 6 video. video memiliki *framerate* yang sama yaitu 25. Daftar input uji coba tersebut antara lain disajikan dalam Tabel 5.1.

Tabel 5.1 Data video yang digunakan

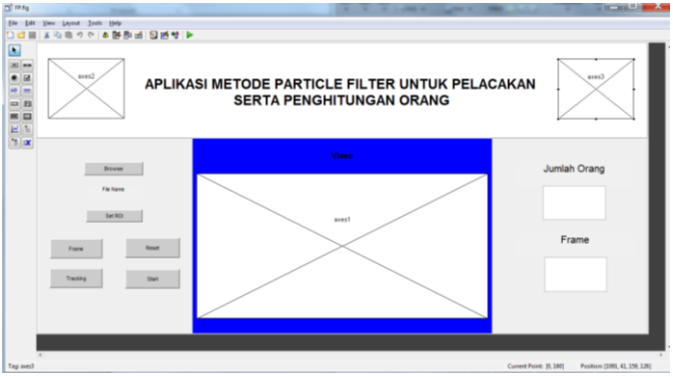
No	Nama	*FPS*	Pixel
1.	Video1.avi	25	640x480
2.	Video2.avi	25	640x480
3.	Video3.avi	25	640x480
4.	Video4.avi	25	640x480
5.	Video5.avi	25	640x480
6.	Video6.avi	25	640x480

5.2 Graphical User Interface Program

Salah satu aspek penting dalam pembuatan program adalah perancangan antar muka, karena perancangan antar muka yang baik berbanding lurus dengan tingkat *user friendly* sebuah program. Artinya perangkat lunak dirancang dengan sedemikian rupa agar pemakai dapat beradaptasi dengan mudah dalam pemakaian program tersebut. Berikut adalah desain antar muka halaman utama :

Perancangan antar muka halaman utama ditunjukkan pada Gambar 4.2. Form ini adalah form pengerjaan metode *Particle Filter*. Pada halaman ini dilakukan proses pemilihan video, pra pengolahan video, pemilihan ROI. Halaman ini menampilkan video hasil pelacakan objek. Antar muka halaman utama terdiri dari :

1. Axes1, berfungsi untuk menampilkan frame pertama dari video input.
2. Push button Open File, berfungsi untuk memilih video input yang akan digunakan dalam proses pelacakan.
3. Push button Set ROI, berfungsi untuk mengatur daerah yang ingin dilacak serta dihitung.
4. Push button Reset digunakan untuk menghapus semua historis dari proses sebelumnya.
5. Push button Tracking, berfungsi untuk memulai proses pelacakan serta penghitungan orang.
6. Push button Stop, berfungsi untuk menghentikan proses pelacakan serta penghitungan orang.
7. Edit text Jumlah Orang, berfungsi agar menampilkan jumlah orang yang berada dalam area ROI.
8. Edit text Frame, berfungsi agar menampilkan jumlah frame yang telah berjalan.



Gambar 5.1 GUI Program

5.3 Lingkungan Pengujian Prototipe Perangkat Lunak

Lingkungan pengujian dari prototipe perangkat lunak pelacakan serta penghitungan orang meliputi perangkat keras dan lunak komputer. Detail dari perangkat keras dan lunak yang digunakan dapat dilihat pada Tabel 5.2

Tabel 5.2 Lingkungan Pengujian Prototipe Perangkat Lunak.

Perangkat keras	Intel Core i5 processor 2400S 2.5GHz Memory 4096MB RAM
Perangkat lunak	Sistem operasi Microsoft Windows 7 Pro 64 bit Tools menggunakan MATLAB R2017a.

5.4 Pengujian Tahap Pelacakan Orang

Pengujian prototipe perangkat lunak menggunakan 5 data video. Sebelum melakukan pelacakan objek dengan metode *Adaptive Particle Filter*, data video terlebih dahulu melalui proses pra pengolahan. Pra pengolahan video yang dilakukan pada Tugas Akhir ini adalah proses ekualisasi histogram untuk setiap frame video. Dari proses ekualisasi histogram didapatkan semua frame video yang memiliki penyebaran histogram yang merata sehingga setiap derajat keabuan memiliki jumlah pixel yang relatif sama. Frame yang diolah selama proses pelacakan objek adalah frame hasil ekualisasi histogram.

Proses pra pengolahan video dilanjutkan dengan proses pelacakan orang. Pada proses pelacakan orang, dibutuhkan vektor gerakan dari orang untuk ditambahkan pada model ruang keadaan. Vektor gerakan orang diperoleh dengan menggunakan metode *Hierarchical Block Matching*. Pelacakan orang dengan metode *Particle Filter* menggunakan sejumlah partikel dengan bobotnya masing-masing. Kemudian bobot akan mengalami perubahan sesuai dengan fungsi likelihood yang didefinisikan. *Resample*

partikel juga dapat dilakukan jika nilai *effective sample size* kurang dari setengah jumlah partikel. Selanjutnya adalah proses estimasi yang dilakukan dengan menghitung rata-rata bobot. Jika orang terdeteksi pada waktu t-1, maka pada waktu t proses estimasi dilakukan dengan menghitung rata-rata bobot. Sedangkan bila orang tidak terdeteksi pada waktu t-1, maka pada waktu t proses estimasi dilakukan dengan mengambil bobot maksimum kemudian menghitung rata-rata bobot.

Diambil contoh dari frame ke-6 pada video Video1.avi, setelah didapatkan frame maka akan dilakukan ekualisasi histogram, dimana diambil dari pixel baris ke 500 dan kolom 100 dengan ukuran 5x5 didapatkan nilai pixel:

$$r = \begin{bmatrix} 126 & 127 & 127 & 127 & 128 \\ 126 & 129 & 136 & 139 & 137 \\ 133 & 132 & 130 & 133 & 131 \\ 137 & 136 & 136 & 141 & 129 \\ 137 & 132 & 131 & 131 & 128 \end{bmatrix}$$

$$g = \begin{bmatrix} 125 & 126 & 130 & 131 & 134 \\ 125 & 128 & 137 & 142 & 142 \\ 132 & 131 & 135 & 139 & 139 \\ 137 & 138 & 140 & 148 & 139 \\ 137 & 136 & 137 & 139 & 139 \end{bmatrix}$$

$$b = \begin{bmatrix} 130 & 132 & 137 & 137 & 141 \\ 130 & 135 & 144 & 151 & 150 \\ 135 & 136 & 136 & 147 & 150 \\ 139 & 139 & 144 & 163 & 154 \\ 144 & 142 & 145 & 157 & 161 \end{bmatrix}$$

Kemudian setelah dilakukan ekualisasi histogram didapatkan hasil pixel seperti berikut:

$$gr = \begin{bmatrix} 133 & 133 & 130 & 139 & 159 \\ 129 & 138 & 130 & 131 & 135 \\ 122 & 133 & 141 & 135 & 134 \\ 126 & 129 & 134 & 136 & 123 \\ 128 & 132 & 133 & 132 & 125 \end{bmatrix}$$

Dari pixel diatas terlihat bahwa setelah dilakukan ekualisasi histogram nilai pixel berada diantar 120 hingga 141, ini menunjukkan bahwa ekualisasi histogram membuat *pixel* yang sebelumnya memiliki selisih jauh menjadi cukup dekat dan memliki tingkat keabuan yang cukup tinggi.

Penghitungan intensitas dihitung berdasarkan *likelihood* seperti pada Persamaan 2.4 berdasarkan kemiripan objek yang dilacak T dengan persekitaran partikel I pada frame t menggunakan *Sum of Square Differences* (SSD). Dimisalkan telah memiliki 2 matriks dari *template* dan *image* sebagai berikut.

$$template = \begin{bmatrix} 2 & 3 & 3 \\ 2 & 2 & 2 \\ 1 & 1 & 2 \end{bmatrix} \text{ dan } Image = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

Kedua matrik tersebut akan dihitung menggunakan SSD seperti pada persamaan 2.6 . Sebuah persekitaran kecil dari partikel atau $Neib$ berukuran $M - 1 \times N - 1$ seperti Persamaan 2.6 dapat tuliskan kembali menjadi

$$r(X_t) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [T(i,j) - I(x+i, y+j)]^2 \qquad (2.6)$$

$$r(X_t) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [T(i,j) - I(x+i, y+j)][T(i,j) - I(x+i, y+j)]$$

$$r(X_t) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} T(i,j)^2 - 2 \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} T(i,j) I(x+i, y+j) + \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(x+i, y+j)^2$$

Untuk posisi partikel di x=0; y=0 didapat

$$\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} T(i,j)^2 = 2^2 + 3^2 + 3^2 + \cdots + 1^2 + 2^2 = 40$$

$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} T(i,j)I(x+i,y+j)$$
$$= (2x1) + (3x2) + (3x2) + \cdots + (1x1) + (2x1) = 28$$
$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} I(x+i,y+j)^2 = 1 + 2^2 + 2^2 + \cdots + 1 + 1 = 21$$

sehingga $r(x,y) = 40 - (2x28) + 21 = 5$

Dari perhitungan menggunakan SSD didapatkan hasil sebagai berikut:

- Nilai kemiripan masing-masing partikel $r = \{184,152,192,169,130\}$
- Jumlah kandidat masing-masing partikel $J = \{1,3,2,1,1\}$
- Probabilitas prior $q_j = \frac{1}{\sum_{i=0}^{N-1} J_i} J_i = \left\{\frac{1}{16}, \frac{3}{16}, \frac{2}{16}, \frac{1}{16}, \frac{1}{16}\right\}$
- Faktor normalisasi
$$C_N = \frac{1}{\sum_{i=0}^{N-1} r_i} = \left[\frac{1}{\{184 + 152 + 192 + 169 + 130\}}\right] = 0{,}00121$$
- Sebuah nilai terdistribusi uniform $U(.) = 0{,}5469$

maka dengan menggunakan fungsi likelihood pada persamaan 2.7 didapatkan

$$P\left(Z_t^{int} \middle| X_t\right) = q_0 U(.) + C_N \sum_{j=1}^{J} q_j N(r_t, \sigma_t)$$
$$= 0{,}5(0{,}5469) + 0{,}00121\left[\frac{1}{16}187{,}57 + \frac{3}{16}154{,}76 + \frac{2}{16}192{,}72 + \frac{1}{16}168{,}9 + \frac{1}{16}130{,}7\right]$$
$$= 0{,}5(0{,}5469) + 0{,}00121[83{,}5]$$
$$= 0{,}27345 + 0{,}10103$$
$$= 0{,}37448$$

didapatkan nilai likelihood untuk pengukuran intensitas sebesar 0,37448

Misalkan kondisi pada frame ke-7 objek dapat dideteksi pergerakannya dan diketahui posisi objek sebelumnya adalah sebagai berikut.

Tabel 5.3 Koordinat objek yang diketahui.

Frame	Koordinat pada sumbu-x	Koordinat pada sumbu-y
1	286	117
2	282	117
3	281	117
4	277	117
5	270	120
6	266	121

Proses penghitungan pada frame ke-7 atau t = 7 dijalankan dengan mengetahui nilai $\overline{\Delta x}$ dan $\overline{\Delta y}$ dari perbedaan posisi partikel lama dengan partikel baru seperti persamaan 2.5. Apabila perbedaan posisi partikel lama dan partikel baru diketahui sebagai $\Delta x_t = -6; \Delta y_t = -2, \sigma_{mot} = 1$ dan banyaknya koordinat posisi objek pada frame sebelumnya yang dibutuhkan, misal k=4, maka

Untuk $t = 7, k = 4$ didapatkan

$$\overline{\Delta x} = \sum_{s=t-k}^{t-1} \frac{|x_s - x_{s-1}|}{k}$$

$$\overline{\Delta x} = \sum_{s=3}^{6} \frac{|x_s - x_{s-1}|}{k}$$

$$\overline{\Delta x} = \frac{|x_3 - x_2| + |x_4 - x_3| + |x_5 - x_4| + |x_6 - x_5|}{4}$$

$$\overline{\Delta x} = \frac{|281 - 282| + |277 - 281| + |270 - 277| + |266 - 270|}{4}$$

$$\overline{\Delta x} = \frac{1+4+7+4}{4} = 4$$

dan

$$\overline{\Delta y} = \sum_{s=t-k}^{t-1} \frac{|y_s - y_{s-1}|}{k}$$

$$\overline{\Delta y} = \sum_{s=3}^{6} \frac{|y_s - y_{s-1}|}{k}$$

$$\overline{\Delta y} = \frac{|y_3 - y_2| + |y_4 - y_3| + |y_5 - y_4| + |y_6 - y_5|}{4}$$

$$\overline{\Delta y} = \frac{|117-117| + |117-117| + |120-117| + |121-120|}{4}$$

$$\overline{\Delta y} = 1$$

Kemudian menghitung perubahan posisi partikel dengan kecepatan rata-rata objek pada waktu sebelumnya berdasarkan persamaan 2.8

$$d_{mot}^2 = (|\Delta x_t| - \overline{\Delta x})^2 + (|\Delta y_t| - \overline{\Delta y})^2$$

$$d_{mot}^2 = (|6| - |4|)^2 + (|2| - |1|)^2$$

$$d_{mot}^2 = (2)^2 + (1)^2 = 5$$

Setelah mendapatkan hasil dari perhitngan diatas akan digunakan untuk mendapatkan nilai pengukuran gerakan dengan menggunakan fungsi *likelihood* pada persamaan 2.8

$$P(Z_t^{mot}|X_t) = \frac{1}{\sqrt{2\pi}\sigma_{mot}} \exp\left(-\frac{d_{mot}^2}{2\sigma_{mot}^2}\right)$$

$$P(Z_t^{mot}|X_t) = \frac{1}{\sqrt{2\pi}.1} \exp\left(-\frac{2}{2.5.5}\right)$$

$$P(Z_t^{mot}|X_t) = 0{,}083$$

sehingga didapatkan *likelihood* untuk pengukuran gerakan sebesar 0,083

Intensitas trayektori dihitung berdasarkan kedekatan partikel terhadap lintasan yang diperoleh dari posisi objek pada *frame* sebelumnya berdasarkan persamaan:

$$y = \sum_{i=0}^{m} a_i x^i$$

Koefisien tersebut dapat dicari dengan metode linear sehingga ditemukan nilai koefisiennya dengan nilai koefisiennya dengan nilai $a_2 = 0; a_1 = 8{,}3\ x\ 10^{-6}; a_0 = 1{,}2947\ x\ 10^{-3}$. Sebuah partikel pada posisi $x = 281; y = 117$ akan digunakan untuk mencari nilai closeness metric berdasarkan koefisien yang telah didapatkan pada proses diatas, didapatkan

$d_{trj} = \left|y - \sum_{i=0}^{m} a_i x^i\right|$

$d_{trj} = |y - (a_0 x^0 + a_1 x^1 + a_2 x^2)|$

$d_{trj} = |117 - (1{,}2947\ x\ 10^{-3} + (8{,}3\ x\ 10^{-6}) x\ 281) + 117^2 x\ 0|$

$d_{trj} = 0{,}003627$

Dua frame sebelumnya dimisalkan tidak terdeteksi gerakannya, dengan $\lambda_f = 0{,}9$ dan $\sigma_{trj}^2 = 1$, didapatkan nilai pengukuran trayektorinya, sebagai berikut

$$P\left(Z_t^{trj} \middle| X_t\right) = \frac{1}{\sqrt{2\pi}\sigma_{trj}} \exp\left(-\frac{\frac{d_{trj}}{F^2}}{2\sigma_{trj}^2}\right)$$

$$P\left(Z_t^{trj} \middle| X_t\right) = \frac{1}{\sqrt{2\pi}\ x\ 1} \exp\left(-\frac{0{,}003627/0{,}9^2}{2\ x\ 1\ x1}\right)$$

$P\left(Z_t^{trj} \middle| X_t\right) = 0{,}00159$

Nilai *likelihood* untuk pengukuran trayektori sebesar 0,00159

Setelah bobot diperbarui, akan dipilih bobot terbaik pada proses resampling apabila nilai *effective sample size* kurang dari setengah jumlah partikel. Kemudian posisi terbaru akan diperoleh dari rata-rata partikel bobot masing-masing. Namun, apabila estimasi pergerakan tak terdeteksi, maka kita akan mengestimasi posisi objek pada *frame* berikutnya dan memperhalus lintasan

dengan cara memproyeksikan salah satu prediksi posisi benda ke estimasi lintasan.

Sebuah proses estimasi posisi objek pada frame ke-7 akan dilakukan misal dengan kondisi objek tidak terdeteksi pergerakannya ada frame ke-5. Jika diketahui informasi posisi objek pada frame sebelumnya dengan j > i, maka

$$\hat{X}_{cur} = X_j + \frac{(X_j - X_i)(cur - j)}{j - i}$$

$$\hat{X}_{cur} = \begin{bmatrix}264\\125\end{bmatrix} + \left(\begin{bmatrix}264\\125\end{bmatrix} - \begin{bmatrix}265\\121\end{bmatrix}\right)\frac{1}{2}$$

$$\hat{X}_{cur} = \begin{bmatrix}264\\125\end{bmatrix} + \begin{bmatrix}-1/2\\2\end{bmatrix} = \begin{bmatrix}263{,}5\\127\end{bmatrix} \approx \begin{bmatrix}264\\127\end{bmatrix}$$

Kemudian memproyeksikan $\hat{X}_{cur}$ ke dalam lintasan terdekatnya yaitu $\hat{X}_{cur}$ dengan menghitung jarak Euclidean dari beberapa posisi objek pada frame sebelumnya. Proses estimasi dilanjutkan dengan

$$X_{cur} = \left(1 - \lambda_f^{t_o}\right)\hat{X}_{cur} + \tilde{X}_{cur}\,\lambda_f^{t_o}$$

$$X_{cur} = (1 - 0{,}9^2)\begin{bmatrix}264\\127\end{bmatrix} + \begin{bmatrix}264\\125\end{bmatrix}(0{,}9)^2 \approx \begin{bmatrix}264\\125\end{bmatrix}$$

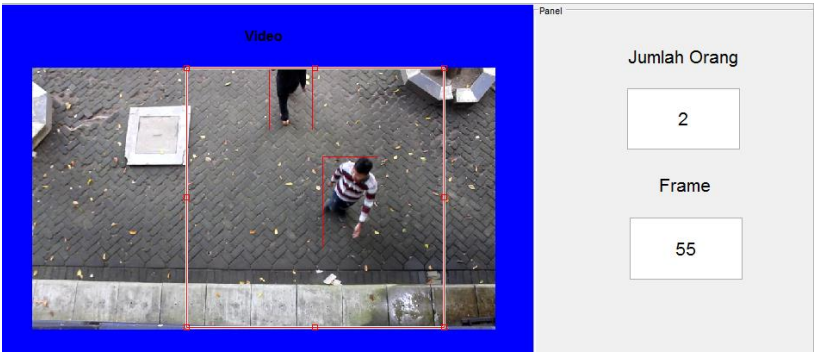
Sehingga diperoleh posisi objek dengan koordinat x=264; y=125

5.5 Uji Data

Dalam sub bab ini dilakukan uji coba program dengan video yang telah direkam sebelumnya. Ada 6 video yang digunakan sebagai bahan uji coba, dengan 3 video sebagai pengujian pelacakan, serta 3 video sebagai pengujian *counting*.

5.5.1 Video 1

Pada video pertama ini digunakan untuk counting orang yang berada dalam area ROI. Ada 2 orang yang berada dalam video, dengan tanpa oklusi , didapatkan hasil yang baik yaitu hanya seperti pada gambar berikut.



Gambar 5.2 Hasil video1.avi

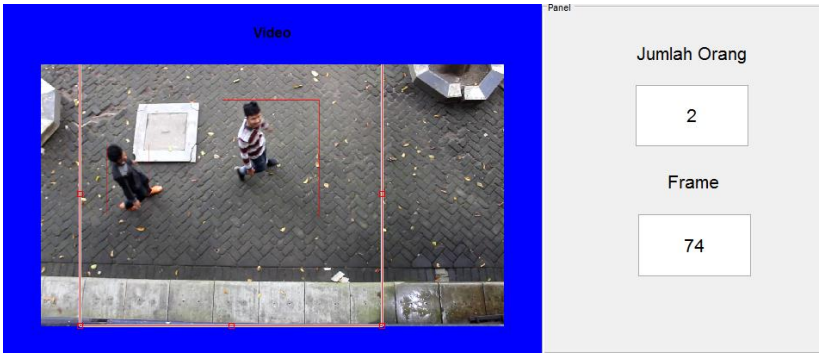
Dari ujicoba pada video pertama didapatkan hasil bahwa program dapat melakukan counting dengan cukup baik, walaupun sesekali ada error dalam countingnya.

Tabel 5.4 Hasil proses Video1.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	71	91 %
	Jumlah Frame Terlacak Salah	7	
Perhitungan	**Jumlah Frame Terhitung Benar**	71	91 %
	Jumlah Frame Terhitung Salah	7	

5.5.2 Video 2

Pada video kedua ini digunakan untuk *counting* orang yang berada dalam area ROI. Ada 2 orang yang berada dalam video, yang berjalan-jalan dengan kondisi tanpa oklusi , didapatkan hasil yang baik yaitu hanya seperti pada gambar berikut.



Gambar 5.3 Hasil video2.avi

Dari ujicoba pada video kedua didapatkan hasil bahwa program dapat melakukan *counting* dengan cukup baik.

Tabel 5.5 Hasil proses Video2.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	58	100 %
	Jumlah Frame Terlacak Salah	0	
Perhitungan	**Jumlah Frame Terhitung Benar**	58	100 %
	Jumlah Frame Terhitung Salah	0	

5.5.3 Video 3

Pada video ketiga ini digunakan untuk *counting* orang yang berada dalam area ROI. Ada 7 orang yang berada dalam video, yang berjalan-jalan dengan kondisi oklusi sebagian, didapatkan hasil yang seperti pada gambar berikut.

Gambar 5.4 Hasil video3.avi

Dari ujicoba pada video ketiga didapatkan hasil yang kurang baik disebabkan oleh oklusi sebagian yang membuat orang menjadi tidak dapat dihitung.

Tabel 5.6 Hasil proses Video3.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	67	69 %
	Jumlah Frame Terlacak Salah	30	
Perhitungan	**Jumlah Frame Terhitung Benar**	67	69 %
	Jumlah Frame Terhitung Salah	30	

5.5.4 Video 4

Pada video keempat ini digunakan untuk pelacakan orang yang berada dalam area ROI, dimana orang tersebut dalam kondisi berlari, didapatkan hasil sebagai berikut.

Gambar 5.5 Hasil video4.avi

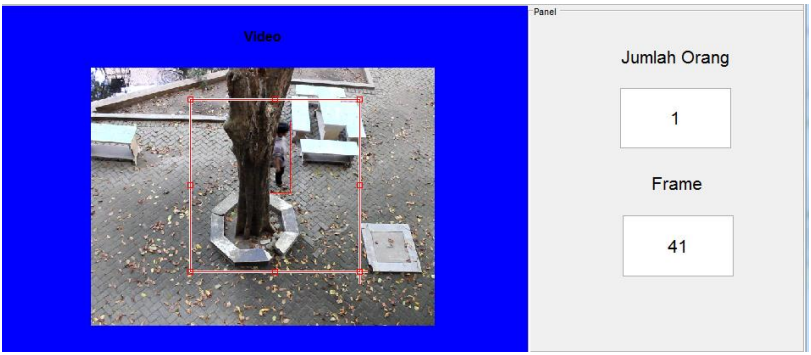
Dari ujicoba pada video keempat didapatkan hasil bahwa program dapat melakukan pelacakan dengan cukup baik, hanya sekali ada error dalam pelacakannya.

Tabel 5.7 Hasil proses Video4.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	25	96 %
	Jumlah Frame Terlacak Salah	1	
Perhitungan	**Jumlah Frame Terhitung Benar**	21	96 %
	Jumlah Frame Terhitung Salah	1	

5.5.5 Video 5

Pada video kelima ini digunakan untuk pelacakan orang yang berada dalam area ROI, dimana orang tersebut dalam kondisi berjalan namun tertutup oleh pohon, didapatkan hasil sebagai berikut.



Gambar 5.6 Hasil video5.avi

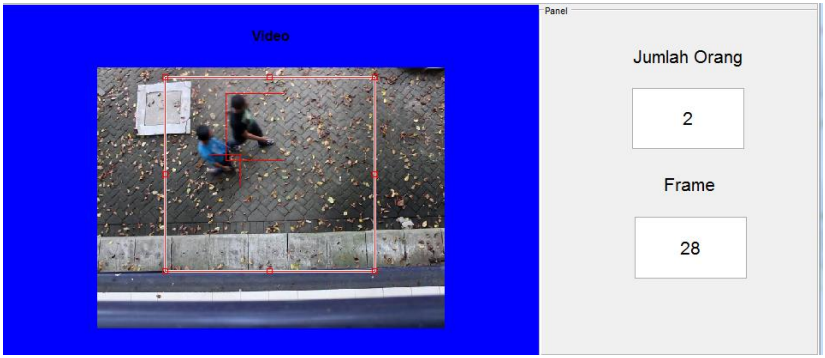
Dari ujicoba pada video kelima didapatkan hasil bahwa program belum dapat melakukan pelacakan dengan baik, karena oklusi yang menyeluruh.

Tabel 5.8 Hasil proses Video5.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	74	88 %
	Jumlah Frame Terlacak Salah	10	
Perhitungan	**Jumlah Frame Terhitung Benar**	74	88 %
	Jumlah Frame Terhitung Salah	10	

5.5.6 Video 6

Pada video terakhir ini digunakan untuk pelacakan orang yang berada dalam area ROI, dimana orang tersebut dalam kondisi berjalan namun secara bersamaan, didapatkan hasil sebagai berikut.



Gambar 5.7 Hasil video6.avi

Dari ujicoba pada video terakhir didapatkan hasil bahwa program dapat melakukan pelacakan namun ketika berjalan bersamaan orang tersebut dianggap satu dikarenakan oklusi sebagian.

Tabel 5.9 Hasil proses Video6.avi

Pelacakan	**Jumlah Frame Terlacak Benar**	35	35 %
	Jumlah Frame Terlacak Salah	65	
Perhitungan	**Jumlah Frame Terhitung Benar**	35	35 %
	Jumlah Frame Terhitung Salah	65	

Dari berbagai video yang telah dilakukan uji coba didapatkan hasil sebagai berikut:

Tabel 5.10 Hasil Video secara keseluruhan

Video	Jumlah Frame	Terdeteksi	Tidak Terdeteksi	Presentase
Video1.avi	78	71	7	91 %
Video2.avi	58	58	0	100 %
Video3.avi	97	67	30	69 %
Video4.avi	26	25	1	96,1 %
Video5.avi	84	74	10	88 %
Video6.avi	100	35	65	35 %

5.6 Pembahasan Penyebab Besar Kecilnya Presentase Kinerja

Penyebab besar kecilnya prosentase kinerja disebabkan oleh beberapa hal sebagai berikut :

1. Berdasarkan uji coba terhadap video2.avi didapatkan hasil bahwa bila tanpa oklusi, kinerjanya untuk pelacakan serta penghitungan sangat baik, hingga mencapai 100 %.
2. Berdasarkan uji coba terhadap video5.avi didapatkan hasil bahwa bila adanya oklusi secara keseluruhan, kinerjanya untuk pelacakan serta penghitungan tidak terlalu baik. Ini disebabkan karena ketika orang melwati pohon, maka orang tersebut tertutupi sehingga tidak terlacak serta terhitung.
3. Berdasarkan uji coba terhadap video6.avi didapatkan hasil bahwa bila adanya oklusi sebagian, pada awalnya dapat dilacak serta dihitung, namun ketika berjalan bersamaan agak berubah, sesekali menjadi 1 objek kembali.
4. Berdasarkan uji coba terhadap video4.avi didapatkan hasil bahwa bila orang yang sedang berlari, kinerjanya untuk pelacakan serta penghitungan sangat baik, hingga mencapai 96 %.

5. Ketika terjadi oklusi secara seluruhnya, orang akan terdeteksi secara acak-acakan, tidak teratur.
6. Ketika terjadi oklusi sebagian pada awalnya dapat melacak dengan baik, namun semakin lama orang akan dihitung satu.

BAB VI
PENUTUP

Bab ini berisi tentang beberapa kesimpulan yang dihasilkan berdasarkan penelitian yang telah dilaksanakan. Disamping itu, pada bab ini juga diberikan beberapa saran yang dapat digunakan jika penelitian ini ingin dikembangkan.

6.1 Kesimpulan

Berdasarkan uji coba dan pembahasan terhadap hasil pengujian yang telah dilakukan terhadap prototipe perangkat lunak untuk pelacakan objek menggunakan metode *Particle Filter*, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Tugas Akhir ini telah berhasil menerapkan metode *Adaptive Particle Filter* untuk melakukan pelacakan dan penghitungan orang pada video orang yang berjalan. Pelacakan dilakukan dengan beberapa tahapan: pemilihan ROI, preprosessing video, estimasi gerak dengan *Block Matching* dan pelacakan serta penghitungan dengan *Particle Filter*.
2. Tugas Akhir ini berhasil mengatasi ketika terjadi oklusi sebagian walaupun masih ada *error* berdasarkan video6.avi yang akurasinya hanya 35 %, sedangkan ketika terjadi oklusi secara keseluruhan, *Particle Filter* belum dapat melacak dengan baik berdasarkan uji coba video5.avi.
3. Selain itu, Tugas Akhir ini berhasil menghitung orang yang berada dalam area ROI walaupun ada *error* yang terjadi, seperti pada video1.avi dan video4.avi dimana masing masing memiliki akurasi sebesar 91 % dan 96 %.

6.2 Saran

Berdasarkan hasil yang dicapai pada penelitian ini, ada beberapa hal yang disarankan untuk perbaikan kinerja dan pengembangan selanjutnya yaitu:

1. Mungkin dapat dicoba pelacakan serta penghitungan objek dengan menggunakan metode yang lain, seperti *Kalman Filter*, *Extended Kalman Filter* dan sebagainya agar dapat dibandingkan metode mana yang menghasilkan hasil yang terbaik.
2. Mungkin dapat dikembangkan lagi penelitian ini untuk penghitngan dengan metode-metode yang lain, seperti HOG, *Neural Networks*, dan lain-lain.

DAFTAR PUSTAKA

[1] Lee, S.L et al. (2014). "*Multiple Object Tracking With Partial Occlusion Handling Using Salient Feature Points*". Information Sciences 278 . Hal. 448–465.

[2] Lefloch, Damien et al. (2008). "*Real-time people counting system using a single video camera*". Real-Time Image Processing Vol. 6811.

[3] E. Amer, E. Dubois, A. Mitiche. (2003). "*Real-time system for high-level video representations: application to video surveillance*". SPIE International Symposium on Electronic Imaging . Hal 530–541.

[4] Y.L. Hou, G.K.H. Pang. (2011). "*People counting and human detection in a challenging situation*", IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. Hal. 24–33.

[5] H. Yang, L. Shao, F. Zheng, L. Wang, Z. Song. (2011). "*Recent advances and trends in visual tracking: a review*", Neurocomputing 74 . Hal. 3823–3831.

[6] L. Jin, J. Cheng, H. Huang.(2010) "*Human tracking in the complicated background by particle filter using color-histogram and HOG*", in: International Symposium on Intelligent Signal Processing and Communication. Hal. 1–4.

[7] C. Chang, R. Ansari. (2005). "*Kernel particle filter for visual tracking*", IEEE Signal Process. Lett. 12 (3). Hal. 242–245.

[8] R. Cabido, A.S. Montemayor, J.J. Pantrigo. (2012). "*High performance memetic algorithm particle filter for multiple object tracking on modern GPUs*", Soft. Comput. 16. Hal. 217–230.

[9] H. Liu, F. Sun. (2012). "*Efficient visual tracking using particle filter with incremental likelihood calculation*". Inf. Sci. 195. Hal. 141–153.

[10] C. Yang, R. Duraiswami, L. Davis.(2005). "*Fast multiple object tracking via a hierarchical particle filter*". IEEE International Conference on Computer Vision (ICCV).

[11] Yu Huang, Joan Llach .“*Tracking the Small Object through Clutter with Adaptive Particle Filter”*.

[12] K. Okuma, A. Taleghani, N. Freitas, J. Little, D. Lowe. (2004). “*A boosted particle filter: multi-target detection and tracking*”, 8th European Conference on Computer Vision (ECCV), Prague, Czech Republic.

[13] Febrinata, Kazis dkk. (2013). “Simulasi dan Analisis *Multiple Object Tracking* Berbasis Citra Dengan Metode *Hierarchical Particle Filter*”. Telkom University.

[14] Zeng, C. and Ma, Ha. (2010). “*Robust Head-shoulder Detection by PCA-Based Multilevel HOG-LBP Detector for People Counting*”. Beijing University of Posts and Telecommunications.

[15] Wulandari, L. (2015). “Pelacakan Objek Kecil Dengan Metode *Adaptive Particle Filter*”.

[16] S. Zhou, R. Chellappa, dan B. Maghaddam. (2004) “*Appearance tracking using adaptive models in a particle filter*”, ACCV.

[17] https://mathworks.com/help/vision/ref/blockmatching.html diakses pada tanggal 19 Juli 2017

[18] M. Djamel, Kheir Eddine. Aziz. (2010) “*Fast People Counting Using Head Detection From Skeleton Graph*”, IEEE International Conference on Advanced Video and Signal Based Surveillance.

[19] Lazuardi, R.A.F. (2014). “Penghitungan Kendaraan Bergerak Berbasis Algoritma Background Subtraction Menggunakan Metode Gaussian Mixture Model”. Tugas Akhir. Jurusan Matematika ITS

[20] Wang, Yao, Ostermann, Jorn, Zhang, Ya-Qin, Ostermann, Joern, dan Zhang Ya-quin. (2001). Video Processing and Communication. Prentice Hall.

[21] Soto, Alvaro. 2001 .“ *Self Adaptive Particle Filter*”.

LAMPIRAN

A1. Kode Fungsi Untuk *SSD* setiap partikel

```
function [ J , stdIT , r ] = SSD( frame,
template, particle, N, neib)
J = 1;
stdIT = 1;
r = 1;
particle = round(particle);
    for(l=1:N)
        for (i= -neib :1:-1)
            if (particle(1,l)+i > 0 &&
particle(2,l)+i > 0 && particle(1,l)-i <
size(frame,2) && particle(2,l)-i <
size(frame,1))
                input_image = frame
(particle(2,l)+i:particle(2,l)-
i,particle(1,l)+i:particle(1,l)-i);
                [N_SSD, var,
SSD]=SSDscore(template,input_image);
                [x,y] = find(N_SSD>0.89);
                J(l) = length(x);
                stdIT(l) = var;

[x,y]=find(N_SSD==max(N_SSD(:)));
                r(l) = SSD(x,y);
                break
            end
        end
    end

function [N_SSD, stdT , SSD] = SSDscore
(T,I,IdataIn)
if(nargin<3), IdataIn=[]; end
[N_SSD, stdT , SSD]=template_match(T,I,IdataIn);

function [N_SSD, stdT ,
SSD]=template_match(T,I,IdataIn)
```

```
T_size = size(T); I_size = size(I);
outsize = I_size + T_size-1;
FT = fft2(rot90(T,2),outsize(1),outsize(2));
Idata.FI = fft2(I,outsize(1),outsize(2));
Icorr = real(ifft2(Idata.FI.* FT));
Idata.LocalQSumI= local_sum(I.*I,T_size);

QSumT = sum(T(:).^2);

% SSD antara template dan citra
SSD=Idata.LocalQSumI+QSumT-2*Icorr;

% Normalisasi pada range 0..1
N_SSD=SSD-min(SSD(:));
N_SSD=1-(N_SSD./max(N_SSD(:)));

% Menghapus Padding
N_SSD=unpadarray(N_SSD,size(I));
Idata.LocalSumI= local_sum(I,T_size);
stdT=sqrt(numel(T)-1)*std(T(:));


function B=unpadarray(A,Bsize)
Bstart=ceil((size(A)-Bsize)/2)+1;
Bend=Bstart+Bsize-1;
B=A(Bstart(1):Bend(1),Bstart(2):Bend(2));


function local_sum_I= local_sum(I,T_size)
% menambahkan padding
B = padarray(I,T_size);
    s = cumsum(B,1);
    c = s(1+T_size(1):end-1,:)-s(1:end-
T_size(1)-1,:);
    s = cumsum(c,2);
    local_sum_I= s(:,1+T_size(2):end-1)-
s(:,1:end-T_size(2)-1);
```

A2. Kode Fungsi untuk *Resampling*

```
function [ indx ] = resample(w)
N = length(w);
M = length(w);
w = w / sum(w);
indx = zeros(1, N);
Ns = floor(N.* w);
R = sum(Ns);
i = 1;
j = 0;
while j < M
    j = j + 1;
  cnt = 1;
  while cnt <= Ns(j)
    indx(i) = j;
    i = i + 1; cnt = cnt + 1;
  end;
end;
N_rdn = N - R;
Ws =(N*w - Ns)/N_rdn;
Q = cumsum(Ws);
while(i <= N)
    sampl = rand;
    j = 1;
    while(Q(j) < sampl),
        j = j + 1;
    end;
    indx(i) = j;
    i = i + 1;
end
```

"Halaman ini sengaja dikosongkan"

B.1 Kode Fungsi Estimasi Gerak dengan metode *Hierarchical Block Matching*

```
function [ motion ] = Hierarchical( Ref_Img,
Target_Img, MSize,sr, loct)

[height ,width]=size(Target_Img);

Ref_Img1 = imresize(Ref_Img, 0.5);
Ref_Img2 = imresize(Ref_Img1, 0.5);

Target_Img1 = imresize(Target_Img, 0.5);
Target_Img2 = imresize(Target_Img1, 0.5);

L=3;
Factor=2.^(L-1);
sr= round(sr/Factor);
height = height/Factor;
width = width/Factor;

%compute for 1st Level
block = floor(loct/MSize);
block = (block*MSize)+1-(3*MSize);

if (block(1)<=0)
    block(1)=1;
end
if (block(2)<=0)
    block(2)=1;
end

[motion] = motionEstES(Ref_Img2,
Target_Img2,MSize,sr,block);

%for the next level

for ii=L-1:-1:1
```

```
    vectors = zeros(4,4);

        if ii==1
              imgP=Ref_Img(:,:);
              imgI=Target_Img(:,:);
              blok = block;
         else if ii==2
              imgP=Ref_Img1(:,:);
              imgI=Target_Img1(:,:);
              blok =
uint32(block/2/MSize)*MSize+1 ;
             end
        end
%
   %Update all parameters for the current level.
   motion = motion*2;
   height = height*2;
   width = width*2;
   row = height;
   col = width;
   ttt = size(motion);
   costs = ones(2*sr + 1, 2*sr +1) * 65537;
   computations = 0;
   s = sr;

    for c = 4:-1:0
        a = blok(2)+(c*MSize);
        if (a<=height)
            row = a;
            break;
        end
    end

   for c = 4:-1:0
        b = blok(1)+(c*MSize);
        if (b<=width)
            col = b;
            break;
```

```
        end
    end
    index = uint32(blok/MSize)+1;
    if(mod(index(1),2)==0)
        q = 1;
    else
        q = 2;
    end

    if(mod(index(2),2)==0)
        l = 1;
    else
        l = 2;
    end

     mbCount = 1;
     mbRow = 1;

     for i=block(2):MSize:row
       baseline=(uint32(q/2));
       for k=block(1):MSize:col
          mindx=(uint32(l/2));
          if mindx>ttt(2)
             mindx=ttt(2);
          end

          if baseline>ttt(1)
             baseline=ttt(1);
          end

          y = i+imag(motion(baseline,mindx));
          x = k+real(motion(baseline,mindx));

          for m = -s : s
             for n = -s : s
                 refBlkVer = y + m;   % row/Vert
co-ordinate for ref block
```

```
                refBlkHor = x + n;   %
col/Horizontal co-ordinate
                if ( refBlkVer < 1 ||
refBlkVer+MSize-1 > height ...
                        || refBlkHor < 1 ||
refBlkHor+MSize-1 > width || k+MSize-1 >
width...
                        || i+MSize > height)
                    continue;
                end

 costs(m+s+1,n+s+1) =
costFuncMAD(imgP(i:i+MSize-1,k:k+MSize-1), ...

imgI(refBlkVer:refBlkVer+MSize-1,
refBlkHor:refBlkHor+MSize-1), MSize);
                computations = computations + 1;
            end
        end

        if (costs == 65537)
            dy = s+1;
            dx = s+1;
        else
        [dx, dy, min] = minCost(costs);
        end

vectors(1) =(dy-s-
1)+imag(motion(baseline,mindx));
vectors(2) = (dx-s-
1)+real(motion(baseline,mindx));
        vectors(mbRow,mbCount) =
[vectors(2)+vectors(1)*j];
        costs = ones(2*s + 1, 2*s +1) * 65537;

     mbCount = mbCount + 1;
   l= l+1;
end
    l=1;
```

```
    q = q+1;
    mbRow = mbRow + 1;
    mbCount = 1;
    end
   motion = vectors;
end

function [motionVect] = motionEstES(imgP, imgI,
mbSize, p, block)
 [height,width] = size(imgI);

 if nargin < 5
     vectors =
zeros(round(height/mbSize),round(width/mbSize));
     block = [1 1];
     row = height;
     col = width;
 else
    block =
uint32((uint32(block/2/mbSize)*mbSize+1)/2/mbSiz
e)* mbSize+1;
    vectorRow = 1:mbSize:height ;
    vectorCol = 1:mbSize:width ;

    if (block(1)> vectorCol(end))
         block(1)=vectorCol(end);
    end
    if (block(2)> vectorRow(end))
          block(2)=vectorRow(end);
    end

    row = height;
    col = width;
    for c = 4:-1:0
        a = block(2)+(c*mbSize);
        if (a<=height)
            row = a;
            break;
        end
```

```
    end

   for c = 4:-1:0
        b = block(1)+(c*mbSize);
        if (b<=width)
            col = b;
            break;
        end
   end
  end

costs = ones(2*p + 1, 2*p +1) * 65537;
computations = 0;

mbCount = 1;
mbRow = 1;
for i = block(2): mbSize : row
    for k = block(1) : mbSize : col

        for m = -p : p
            for n = -p : p
                refBlkVer = i + m;
                refBlkHor = k + n;
                if ( refBlkVer < 1 ||
refBlkVer+mbSize-1 > height ...
                        || refBlkHor < 1 ||
refBlkHor+mbSize-1 > width || k+mbSize-1 >
width...
                        || i+mbSize-1 > height)
                    continue;
                end
costs(m+p+1,n+p+1) =
costFuncMAD(imgP(i:(i+mbSize-1),(k):(k+mbSize-
1)), ...

imgI((refBlkVer):(refBlkVer+mbSize-
1),(refBlkHor):(refBlkHor+mbSize-1)), mbSize);
                computations = computations + 1;
            end
```

```
        end

        if (costs == 65537)
            dy = p+1;
            dx = p+1;
        else
        [dx, dy, min] = minCost(costs);
        end

        vectors(1) = dy-p-1;
        vectors(2) = dx-p-1;

vectors(mbRow,mbCount) =
[vectors(2)+vectors(1)*j];
        costs = ones(2*p + 1, 2*p +1) * 65537;
        mbCount = mbCount + 1;
end
    mbRow = mbRow + 1;
    mbCount = 1;
end

motionVect = vectors;

function cost = costFuncMAD(currentBlk,refBlk,
n)
err = 0;
for i = 1:n
    for j = 1:n
        err = err + abs((currentBlk(i,j) -
refBlk(i,j)));
    end
end
cost = err / (n*n);

function [dx, dy, min] = minCost(costs)

[row, col] = size(costs);
min = 65537;
```

```
for i = 1:row
    for j = 1:col
        if (costs(i,j) < min)
            min = costs(i,j);
            dx = j; dy = i;
        end
    end
end
```

B.2 Kode Fungsi Pelacakan Orang dengan Metode *Particle Filter*.

```
function [trackedLocation] = adaptivePF ( video,
roi, mbSize , search, particle ,frame)
  global obj frame1 frame2 prevFrame  Detect
label...
   X  Xt t occ j w r varians;
  trackedLocation = [];
  Detect = [];
  frame1 = [];
  frame2 = [];
  prevFrame = [] ;
  X = [];
  Xt = [];
  w = [];
  r = [];
  obj.videoReader =
vision.VideoFileReader(video);
  obj.videoPlayer =
vision.VideoPlayer('Position',
[100,100,700,400]);
  t = 0;
  occ = 0;
  j = 1;
  varians = 2;
while ~isDone(obj.videoReader)
    t = t+1;
    citra = readFrame();
```

```
    frame1 = frame(:,:,:,t);
    frame2 = rgb2gray(frame1);

      if t==1
      template = imcrop(frame2,roi);
      cent = [roi(1)+uint32(roi(3)/2)
roi(2)+uint32(roi(4)/2)];
             trackedLocation(t,:)=cent
             Detect(j,:)=trackedLocation(t,:);

           %% generate particle at time t-1
            N = particle;%number of particle
            ON = ones(1 , N);
            X3 = zeros(2, N);
            cte = 1/N;
            cteN = cte(1 , ON);
w = cteN;
X =
bsxfun(@plus,trackedLocation(t,:)',sqrt(varians)
*randn(2,N));
Xt = X;
label = 'initial';
track = [(trackedLocation(t,1)-(roi(3)/2))+1
(trackedLocation(t,2)-(roi(4)/2))+1];
regionTrack = [track roi(3) roi(4)] ;
annotateTrackedObject(regionTrack);
point(j,1) = t;
prevFrame = frame2;

      else
            X = bsxfun(@plus,trackedLocation(t-
1,:)',sqrt(varians)*randn(2,N));
            [motion] =
Hierarchical(prevFrame,frame2,mbSize,search,trac
kedLocation(t-1,:));
motion = motion(2:end,2:end);
c =(find(motion==max(motion(:))));
u = real(motion(c(1)));
v = imag(motion(c(1)));
```

```
Xt = repmat (X,1,1);
    if (u~=0 || v~=0)
       pixel = round(trackedLocation(t-1,:));
       Ix = prevFrame(pixel(2),pixel(1)+1)-
prevFrame(pixel(2),pixel(1));
       Iy = prevFrame(pixel(2)+1,pixel(1))-
prevFrame(pixel(2),pixel(1));
       It = frame2(pixel(2),pixel(1)) -
prevFrame(pixel(2),pixel(1));
        miu = abs((u*Ix)+(v*Iy)+It);
        Xt(1,:)= normrnd(X(1,:)+u,miu);
        Xt(2,:)= normrnd(X(2,:)+v,miu);
    else
        Xt(1,:)= normrnd(X(1,:)+u,1);
        Xt(2,:)= normrnd(X(2,:)+v,1);
    end
 [ J , stdIT , r ] = SSD (frame2,template, Xt ,N
, mbSize);

a = 0; b = 1;
q0 = 0.5;
C = 1/sum(r);

for i= 1:N
   qj = (1-0.5)/J(i);
   p = qj*(normrnd(r(i),stdIT(i),[1 J(i)]));
   P1(i)= (q0*unifrnd(a,b))+(C*sum(p));
end

    if (u~=0 || v~=0)
      %average object speed
       delta2 = 0;
        for o=-30:-1
         if(t+o<=0)
           Xs = [0 0];
            else
           Xs = trackedLocation(t+o,:);
            end
```

```
            if(t+o-1<=0)
                Xs1 = [0 0];
                else
                Xs1 = trackedLocation(t+o-1,:);
            end
               delta2_0 = Xs - Xs1;
               delta2 = delta2 + delta2_0;
            end

            delta2 = delta2/30 ;
            %particle speed
 for i=1:N
 delta1 =[ abs(X(1,i)-Xt(1,i)) abs(X(2,i)-
Xt(2,i))];
 d = ((delta1(1)- delta2(1))^2+ (delta1(2)-
delta2(2))^2);
 P2(i)= (1/sqrt(2*pi))*(exp(-1/2 *(d/(1.^2))));
 end
 w = P1.*P2;
 j=j+1;
 point(j,1)= t;

%% Normalize weight vector
   wk = w./sum(w);

 %% Calculate effective sample size
    Neff = 1/sum(wk.^2);
    resample_percentaje = 0.50;
    Nt = resample_percentaje*N;
    if Neff < Nt
       disp('Resampling ...\n')
       [index] = resampleResidual(w);
        Xt = Xt(:,index);
        w=ones(1,N)./N;
        end

   % weighted average
     Detect (j,:) =
(sum(bsxfun(@times,Xt,w),2)/sum(w))';
```

```
     trackedLocation(t,:) = Detect(j,:);
      track = [(trackedLocation(t,1)-
(roi(3)/2))+1 (trackedLocation(t,2)-
(roi(4)/2))+1];
      regionTrack = [track roi(3) roi(4)] ;
      annotateTrackedObject(regionTrack);
      prevFrame = frame2;

else
   fprintf('no motion')
   occ = occ+1;
   koef_poly = polyfit
(trackedLocation(:,1),trackedLocation(:,2),2);
   for i=1:N
   poly = [ Xt(2,i)^2 Xt(2,i) 1];
   d_trj = abs (Xt(2,i)-(sum(koef_poly.*poly)));
   P2(i)= (1/sqrt(2*pi))*(exp(-1/2
*((d_trj/(0.9^occ))/1).^2));
   end
    w = P1.*P2 ;
    %% Normalize weight vector
    wk = w./sum(w);

    %% Calculate effective sample size
     Neff = 1/sum(wk.^2);
     resample_percentaje = 0.50;
     Nt = resample_percentaje*N;
       if Neff < Nt
          disp('Resampling ...\n')
          [index] = resample(w);
           Xt = Xt(:,index);
           w=ones(1,N)./N;
      end
 max_weight = find(w==max(w(:)));
 num = size(max_weight, 2);
 if(num > 1)
    for l=1:num
        X_baru(1,l) = Xt(1, max_weight(l));
        X_baru(2,l) = Xt(2, max_weight(l));
```

```
    end
    maks = w(1,max_weight(1));
    trackedLocation(t,:)
=(sum(bsxfun(@times,X_baru,maks),2)/num*maks)';
  else
trackedLocation(t,1) = Xt(1,max_weight);
trackedLocation(t,2) = Xt(2,max_weight);
end

if t==2 || t==3
   cur = trackedLocation(t-1,:);
    closest_point = cur;
    trackedLocation(t,:) =((1-
0.9^occ)*cur)+(closest_point*(0.9^occ));
else
    cur= Detect(j,:)+((Detect(j,:)-Detect(j-
1,:)) * ((t-point(j,1))/(point(j,1)-point(j-
1,1))))
    point_trj = trackedLocation(1:end-1,:);
D = pdist2(point_trj,cur,'euclidean');
    closest = find(D == min(D(:)));
    closest_point =
trackedLocation(closest(1),:);
    trackedLocation(t,:) =((1-
0.9^occ)*cur)+(closest_point*(0.9^occ));
end

track = [(trackedLocation(t,1)-(roi(3)/2))+1
(trackedLocation(t,2)-(roi(4)/2))+1];
regionTrack = [track roi(3) roi(4)] ;
annotateTrackedObject(regionTrack);
prevFrame = frame2;

            end
      end
  end

function citra = readFrame()
      citra = step(obj.videoReader);
```

```
end

function annotateTrackedObject(regionTrack)
       if (isnan(track))
          videoPlayer.step(citra);
       else

       Imf = insertShape(citra, 'Rectangle',
regionTrack, 'Color', 'red');
        obj.videoPlayer.step(Imf);
       end
    end
end
```

BIODATA PENULIS

Muhammad Sifa'ul Rizky, lahir di Tuban, 8 Agustus 1995. Penulis berasal dari Kota Surabaya, bertempat tinggal di Jalan Kandangan Gunung Dharma II/11 RT 04 RW 01 Kec. Benowo Kota Surabaya. Pendidikan formal yang pernah ditempuh yaitu pendidikan dasar di SDN Kandangan I/121 Surabaya sepanjang tahun 2001-2007, dan pendidikan menengah pertama di SMPN 2 Surabaya pada tahun 2007-2010. Dan melanjutkan pendidikan menengah atas di SMAN 21 Surabaya sejak tahun 2010-2013. Mulai menyandang status mahasiswa di Departemen Matematika FMIPA ITS pada tahun 2013 melalui jalur penerimaan SBMPTN dan menyelesaikan studi S1 nya di tahun 2017. Semasa di bangku perkuliahan, penulis mengikuti beberapa organisasi kemahasiswaan dan kepanitiaan, seperti Himpunan Mahasiswa Matematika (Himatika) ITS, serta event Olimpiade Matematika ITS (OMITS). Selain itu penulis juga aktif sebagai asisten lab. Di akhir tahun masa perkuliahan, penulis aktif di Lab. Ilmu Komputer Matematika untuk memperdalami skill berbasis keilmuan *programming*. Penulis dapat dihubungi lebih lanjut di msifaulkiki@gmail.com.